Multi-path Streaming and Dynamic End-point Adaptation

TUNG, Tak Fu

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy
in
Computer Science and Engineering

June, 2002

Abstract

Advances in networking and multimedia technology makes multimedia streaming on the Internet a popular application. However, the quality of multimedia streaming is not satisfactory due to the lack of facility to guarantee enough bandwidth for streaming. Although bandwidth reservation protocols are proposed, such protocols require routers support and are difficult to deploy to existing routers on the Internet due to the requirement of storing state information of individual flow. Therefore, it implies that it would require hardware or software upgrade of existing routers on the Internet.

Instead of doing reservation, we propose a multi-path streaming approach with end-point adaptation that dynamically adjust connection weight according to the path status. Due to the dynamic nature of Internet, it is not possible to have a known static good path from senders to receivers. Moreover, the status of a path may change dynamically during a streaming session. Therefore, instead of pre-determining the best path before transmission, a multi-path scheme is used and connections are adapted to maintain the quality of streaming, usually by avoiding the congested path. The proposed approach is easy to deploy when compared to reservation approaches because router support is not required. The computational requirement to maintain the streaming quality of a session is carried out at the end-points of the Internet. Hence, the proposed approach is readily deployable for existing Internet users.

摘要

先進的網絡和多媒體技術使多媒體串流成為網際網絡上一種普遍的應用。然而，由於缺乏設施對多媒體串流作出足夠帶寬的保証，多媒體串流的質量往往未能令人滿意。雖然帶寬保留協定以被提議作足夠帶寬的保証，但是這類協定需要路由器的特別支援，較難部署於現有在網際網絡的路由器。亦由於這類協定需要存儲每個單獨連線流程的狀態信息，這意味著現有在網際網絡的路由器有需要作硬件或軟件升級來支援帶寬保留協定。

我們提議一種採用非保留協定的多重路徑串流的方法以及動態端點適應來根據路徑狀態調整連線接量，以達到高質量的多媒體串流。由於網際網絡的動態特性，往往不可能預先知道一個從伺服器到客戶機的有利路徑。而且，即使我們知道一個最有利的路徑，這路徑的狀態也可能會在多媒體串流其間變動而影響質素。所以，我們提議不在傳輸之前預先決定單一最佳的路徑，而是提議一份多重路徑串流的方法，使用多重連線連接到多個伺服器，并且採用動態端點適應來根據路徑狀態調整連線接量，以避免被擁塞的路徑影響多媒體串流的質量。與保留協定不同，我們提出的多重路徑方法並不需要路由器的特別支援，比較容易部署。而且，所有維持串流質量的計算都是在連線端點進行，並不需要路在由器中進行。因此，我們提出的多重路徑方法能夠迅速地配備在現有網際網絡用戶上。

ACKNOWLEDGMENTS

I wish to express my gratitude to my supervisor Dr. John Lui for giving continuous support and guidance in research. I would like to thank Dr. Leana Golubchik for giving many helpful advice. I would also like to thank Alex Chow and Adam Lee for their support in the NS simulations.

Contents

List of Figures

Chapter 1

Introduction to Multi-path Streaming and Dynamic End-point Adaptation

Multimedia streaming on Internet, particularly video and audio delivery, is becoming one of most popular and important Internet application. Unlike file transfer on the Internet (e.g. web-browsing, FTP), streaming of multimedia objects faces technical difficulty on the delay and the bandwidth of the connection due to a lack of facility to have the delay and bandwidth performance guarantee. Moreover, most of the network domains forming the Internet are mostly packet-switched network and traffic condition is dynamic and the condition varies from time to time. A burst of packets traffic may arrive at a router and may result in packet losses if the buffer of the router is full. Under this type of network congestion, a client and a server of a video/audio streaming application can do little to maintain the quality of multimedia streaming. Due to the lack of quality-of-service (QoS) guarantee, it is a challenging problem to provide both reliable data delivery as well as certain level of quality-of-service guarantee.

One approach to provide QoS guarantee is to reserve sufficient bandwidth for the connection between a client and a server before the streaming session starts. There are number of protocols designed for bandwidth reservation such as the

Resource Reservation Setup Protocol (RSVP). With reservation protocol, the quality of streaming services can be guaranteed by reserving enough resources (e.g, buffer and packet transmission bandwidth at routers) for each streaming session. However, the main drawback of the reservation approaches is that it is difficult to deploy to existing routers on the Internet since these protocols require the support from the routers. Reservation protocols also face the scalability problem since each router has to remember the state of each flow, and stateful design implies that the router has to allocate enough CPU to process and maintain state, therefore, it is difficult to scale up the scheduling algorithm to manage thousands of multimedia flows.

Apart from reservation approach, other approaches to provide QoS guarantee is to perform some form of "best-path" routing for the data transfer before the streaming session when there is multiple replicated copy on multiple servers [1]. This approach only works well for small file transfer and may not provide the desirable performance guarantee for the multimedia streaming application. The reason is that the network condition is dynamic and time varying so that the best route at the connection initiation instant may become congested at later time, which may affect the long streaming session (e.g. a streaming of soccer match or 2-hour movie).

In this thesis, we propose a deployable and robust method to provide reliable multimedia streaming over best-effort network without relying of in-core router support. In particular, we propose a multi-path streaming approach with adaptive connection to maintain the multimedia streaming quality.

1.1 Multi-path Streaming

Let us provide a general description of the multi-path streaming approach. The multi-path streaming is a client-oriented (or end-points) approach to make multiple connections to different servers simultaneously. Figure 1.1(b) illustrates a

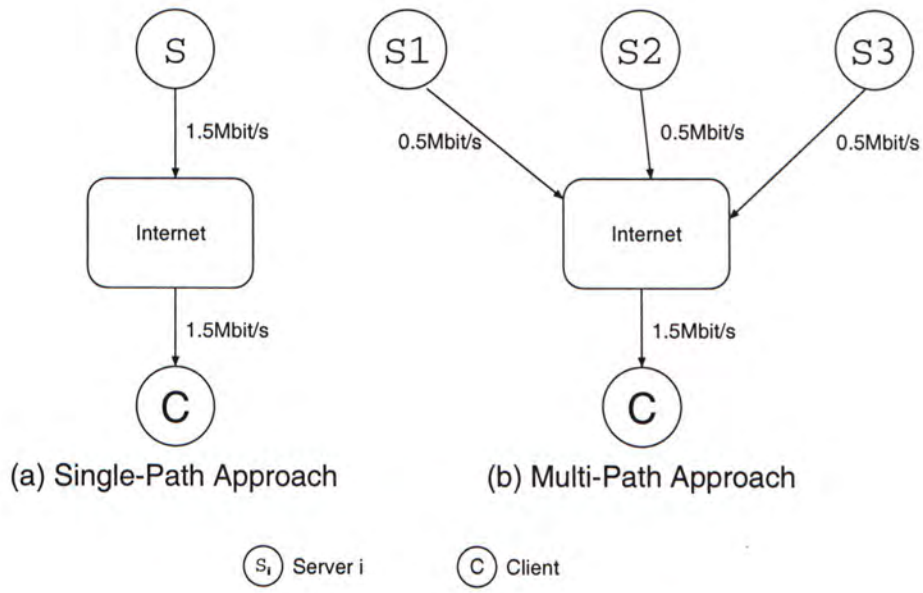


Figure 1.1: Single-path and Multi-path Streaming

configuration of a multi-path streaming application. The client connected to several servers simultaneously and each server stream the data to the client. Note that the streaming is performed in a cooperative way so that no duplicated content will be sent to the network. Thus, the aggregated bandwidth demand to the network is similar to the conventional single-path streaming. One of the possible ways to achieve this cooperation is to stream the data in round-robin manners. For example, the server S_1 in figure 1.1(b) may stream data packets 1, 4, 7, ... and S_2 may stream 2,5,8, ... and S_3 may stream 3,6,9, ... When the multi-path streaming is performed in cooperative manner, the multi-path streaming approach will not cause any extra bandwidth overhead to the network, as compared to the single-path approach that is used by most of the current streaming applications. One important point we need to stress is that the multi-path streaming approach benefits from the increased aggregate bandwidth of server paths. Moreover, multi-path streaming approach can reduce the adverse effect due to network congestion. Figure 1.1(a) shows the single path approach for streaming on the Internet. For example, assume that the server is streaming

an MPEG-I video over the Internet and the sender sends out 1.5 Mbps of data packets. The packets go through the Internet and arrive to the client and the client decodes the received stream. In case a network congestion occurs in the path causing 50% loss of data packets, we will lose 50% of the data. However, for the multi-path streaming approach that is illustrated in Figure 1.1(b), it can reduce such effect of loss due to network congestion. Assuming we are streaming an MPEG-I video, each of the sender will send, for example, 0.5 Mbps of data simultaneously and the receiver will receive a total of 1.5 Mbps of data. If there is a network congestion occurs in the path between sender 1 and the receiver and causes dropping of 50% of data, we only lose 17% of the data instead of 50% as in the single path case. This simple example illustrates the robustness of the multi-path streaming approach can reduce the effect of packet loss due to network congestion. Thus, we believe that multi-path streaming is a promising approach to achieve reliable streaming with QoS guarantee over the Internet. We would also stress the point that the multi-path streaming can be implemented at the application layer and does not rely on special protocol support. So potentially, it can be easily deployed to existing Internet users.

1.2 Dynamic End-point Adaptation

Apart from multi-path streaming, we also address some issues on the dynamic end-point adaptation to achieve reliable streaming with QoS guarantee. Due to the fact that the network condition is highly dynamic, some of the connections may become unreliable during a streaming session. On the other hand, other previously congested link may become more reliable. In this case, the client can switch from the unreliable connection to a more reliable one to maintain the quality.

In general, dynamic end-point adaptation is a scheme to dynamically adjust the amount of delivered traffic on different connections among different servers.

This type of adjustment is necessary when one of the paths becomes congested. Specifically, the client should detect the degradation of quality of the congested route and then switch some traffic from the congested path to the less congested path. For example in figure 1.1(b), client C make a new connection to another server $S4$ when the quality of path 1 is heavily degraded (e.g. dropping 70% of the packets).

The key to the dynamic end-point adaptation is in the measurement of the network status. For example, the client needs to know the available bandwidth of a path, or the parameters describing the characteristics of a path before performing any adaptation. If we are going to do adaptation based on the available bandwidth of the paths, we have to first measure the available bandwidth of each path and try to avoid using the path that has no sufficient bandwidth. Thus we need to have a measurement scheme that can reasonably capture the characteristics of the paths. In the later sections, we present some possible approaches, analytical results as well as result via the model-based simulation.

Chapter 2

Related Work

In this chapter, we will give a brief survey of existing works on the multi-path streaming. We focus on those works that either consider loss characteristics or can be deployed over best-effort service networks, as these are the considerations in our work as well.

Maxemchuk [2, 3, 4] proposed dispersity routing and it is one of the early works that makes use of multiple independent paths for data transfer. The focus of Maxemchuk's work is to reduce the message delay at the network layer. For non-redundant dispersity routing, a message is divided up into submessages of equal length and the submessages are transmitted through different paths of the network. If some of the submessages are lost, the lost submessages are retransmitted while the successfully received submessages are accepted. The message delay is dominated by the number of retransmission needed to receive a message which is determined by the worst path. For redundant dispersity routing, additional redundant submessages are introduced in the message transmission process. With redundant submessages, only a subset of submessages are required to construct the original message. Hence the message delay will be not affected by the worse path. While Maxemchuk's work focus on the delay under a reliable transmission of data messages, our work focus on streaming applications wherein the performance metrics are sustainable transmission bandwidth and to reduce bursty data loss (so as to enhance the perceptual quality of the video). Therefore, there is

a fundamental difference in the performance objective. Under the streaming application, the data is sent through a network at a specific rate (e.g., Mbps or packet per second) and that has an effect on loss characteristics, which we will investigate in later sections. Also, we do not consider retransmissions as there is usually little opportunity to retransmit data in such applications (due to their real-time constraints and continuity requirements). In general, some amount of data lossiness can be tolerated. Another significant difference between our work and the previous work is that we concentrate on application-layer routing rather than the network-layer routing. One major advantage of performing the multi-path routing on the application layer is that we are not relying on the support from the underlying network and this will reduce the deployment difficulty.

Another set of works on the topic considers higher level mechanisms, but requires some assistance from the lower layers and/or assumes significant knowledge of network topology and/or link capacities and delays (on all links used for data delivery). Given such knowledge, algorithms are proposed to select paths which can avoid congested routes. For instance, in [5], the authors focus on adaptation of delivery rate along the different paths, based on losses observed at the receiver. In [6], the authors consider proper scheduling of the initial portion of the video so as to reduce the start-up delay. In contrast, our approach does not rely on specific knowledge of topologies, capacities, delays, etc., and only considers whether a set of paths do or do not share points of congestion, as can be detected at the end-hosts. Moreover, our focus in this thesis is on characterizing the benefits, with respect to loss characteristics, of a multi-path approach as compared to a single path approach. Hence, our interest is in the more basic understanding of using the multi-path approach for data streaming applications.

Recent literature on this topic also includes works on voice-over-IP type applications. For instance, authors in [7, 8] propose a scheme for real-time audio transmission using multiple independent paths between a single sender and a single receiver. For error recovery, multiple description codings(MDC) is used in a

multi-path delivery and a FEC approach is used in a single-path delivery. Using the MDC, multiple descriptions are generated from the source signal and the descriptions are independently encoded. If all the descriptions are received, then the signal can be constructed with full quality. If some of the descriptions are lost, the signal can still be re-constructed but with a degraded quality. These approaches of using MDC and single path with FEC reported in [7, 8] are evaluated through simulation. In contrast, we believe that it is important to understand the effects of multi-path delivery on loss characteristics of streaming video, both for the cases of with or without the use of erasure coding techniques. Hence, we focus the evaluation of multi-path delivery in these two settings. It is important to note that "live" applications (such as voice-over-IP) have different characteristics than pre-recorded applications (as we are considering here in our work). For instance, one such difference is the need to disperse data in real-time, whereas in our case, we can distribute it to the multiple senders ahead of time; this makes application-level implementation simpler and possibly more efficient.

Recently, authors in [9] consider the delivery of pre-recorded video from multiple senders which are distributed across the network. However, this work focuses on a transport protocol as well as on optimization algorithms for (a) rate distribution among the paths (i.e., how much data to send over each path) and (b) packet distribution among the paths (i.e., which packet should be sent over which path), with the objective of minimizing the loss rate at the receiver. In an effort that will appear in the future [10][1] FEC techniques are added (as compared to [9]). Again, distribution algorithms are considered but with the objective of minimizing the probability of irrecoverable error. One main objective of this work is to allocate rate on a path which satisfies the TCP-friendliness criteria. The authors use use an analytical expression to estimate the appropriate amount of bandwidth allocation so to achieve TCP-friendliness property. However, recent

[1]This paper has not appeared yet, and hence we are referring to the version currently available on the authors' web page.

work has shown that a naive application of this approach may not lead to the TCP-friendly property. In contrast, due to the nature of the application, we believe that it is important to consider loss characteristics even when the losses cannot be fully recovered. That is, since we consider delivery of video (which can be displayed even under some losses) in contrast to file transfer (which cannot tolerate losses). In later sections, we also consider data loss rate (with and without the use of FEC), packet loss burst length distribution (with and without the use of FEC), as well as lag-1 autocorrelation (with and without the use of FEC) of packet losses, in our evaluation of potential benefits of multi-path streaming.

Chapter 3

Path Loss Model

A path loss model is necessary for doing analysis. However, due to diverse structure of protocols and applications running on Internet, it is very difficult to find a single congestion model that fit every congestion behavior. Different applications may use different protocols on Internet. A file transfer application may use file transfer protocol that built on top of TCP. A audio streaming streaming application may use UDP with constant sending rate. Even different TCP implementations may have different parameters: different timeout, different window size, etc. Finding a good analytical model for a single protocol is already a very hard problem. It is even harder to have a single analytical model that can accurately model the congestion caused by these diverse protocols running on the Internet.

3.1 Bursty Loss

Many research works have been done to analyze the loss process and conclude that the loss of packets are bursty [11] [12]. That is the loss of consecutive packets for end-to-end traffic are not independent but are correlated. Hence, it is not accurate to simply model the packet losses on the same path as an independent events. We need a loss model that can reasonably capture the real loss behavior for further analysis. We will discuss a 2-state Markov process called Gilbert

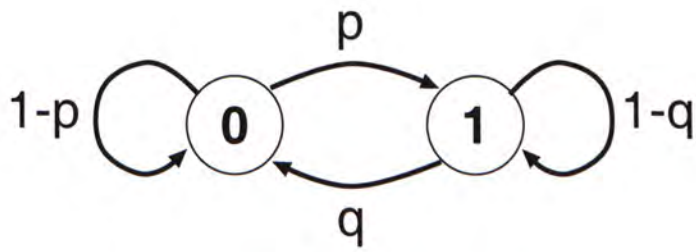


Figure 3.1: The Gilbert Model

model.

3.2 Gilbert Model

3.2.1 Discrete-time Gilbert Model

Let us consider the simpler discrete Gilbert model [13] to model the end-to-end loss behavior. The discrete Gilbert model is a 2-state (1st order) Markov process model that model the 2 states:

- State 0: represent a packet reaching the destination.
- State 1: represent a packet loss.
- p: probability going from state 0 to state 1.
- q: probability going from state 1 to state 0.

The discrete Gilbert model is shown in figure 3.1. Whether a packet transmitted through the path can arrive to the receiver is determined purely by the model state of the path. If a packet is transmitted through the path and the state is 0, the packet can reach the destination. On the other hand, if a packet is transmitted through the path and the state is 1, the packet is dropped. The drop characteristics is determined by the parameters p and q of the model. p is the probability going from state 0 to state 1 and q is the probability going from

state 1 to state 0. In other words, p is the probability that a packet is lost given that previous packet is received and q is the probability that a packet is received given that the previous packet is lost.

Intuitively, higher order Markov chain should give more accurate model of the loss process. Maya Yajnik, Jim Kurose and Don Towsley had conducted experiments on MBone network [14] and analysed on modeling loss process by using different order of Markov chain. Experiment results from [14] show that 1-st order Markov chain model perform reasonably well in many cases when compared to higher order Markov chain model. Thus, instead of using higher order Markov chain which will lead to more complicated analysis, we will use 1-st order Markov chain model (the Gilbert model) for modeling of the loss process.

From the discrete Gilbert model, we can obtain the following equations:

$$\text{P}\{\text{Packet } i \text{ is lost} \mid \text{Packet } i-1 \text{ is received}\} = p \tag{3.1}$$

$$\text{P}\{\text{Packet } i \text{ is received} \mid \text{Packet } i-1 \text{ is lost}\} = q \tag{3.2}$$

$$\text{Loss rate } \pi_1 = \frac{p}{p+q} \tag{3.3}$$

$$\text{Receive rate } \pi_0 = \frac{q}{p+q} \tag{3.4}$$

3.2.2 Continuous-time Gilbert Model

One of the disadvantage of using discrete model is that the effect of different packet spacing cannot be shown. Under the discrete model, packets with spacing of 10ms and packets with spacing of 1 sec will have the same loss characteristics. However, the packets with less spacing are expected to have higher loss correlation than that with more spacing. Thus, the effect of varying the packet spacing, or the sending rate (in terms of packet per second) cannot be reflected by the discrete model.

For multi-path streaming, the spacing of packets transmitted through a path is greater than that of the single path. The spacing is a function of the number of servers, which is $\frac{n}{\lambda}$ second where λ is packet sending rate (packets per second)

Figure 3.2: Packet spacing of single-path and multi-path streaming

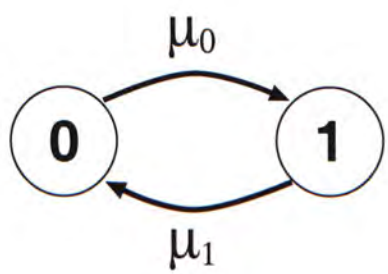


Figure 3.3: Continuous-time Gilbert Model

and n is the number of server. When 2 paths streaming, the packet spacing for both path will be double of the single path one as shown in figure 3.2. For using 3 paths, the spacing will be triple. Thus, we can see that the packet spacing of multi-path streaming is different from that for single path streaming even for the same aggregate sending rate. Discrete Gilbert model may not be appropriate due to the lack of modeling the packet spacing. So we now consider using the continuous-time Gilbert model for the performance analysis.

Using a continuous time Gilbert model, the time-dependent state transition is model as the rate of state change instead of the probability of state change in the discrete case. The continuous-time gilbert model is shown in figure 3.3.

Similar to the discrete model, the continuous-time model has 2 states: state 0 indicates a packet can be successfully received and state 1 indicates a packet is dropped during the transmission. Instead of using probability of state change, for a path k, we have the rate $\mu_0(k)$ which is the rate from state 0 to state 1 and the rate $\mu_1(k)$ which is the rate from state 1 to state 0. The infinitesimal generator for this Gilbert model of path k is:

$$\boldsymbol{Q}_k = \begin{bmatrix} -\mu_0(k) & \mu_0(k) \\ \mu_1(k) & -\mu_1(k) \end{bmatrix}.$$

The stationary distribution of this Gilbert model is

$$\boldsymbol{\pi}(k) = [\pi_0(k), \pi_1(k)]$$

where

$$\pi_0(k) = \frac{\mu_1(k)}{\mu_0(k) + \mu_1(k)}, \text{ and } \pi_1(k) = \frac{\mu_0(k)}{\mu_0(k) + \mu_1(k)}$$

Let $p_{i,j}^{(k)}(\tau)$ be the probability that path k is in state j at time $t + \tau$, given that it was in state i at time t, i.e., $p_{i,j}^{(k)}(\tau) = P(X_k(t+\tau) = j | X_k(t) = i)$. From [15], we have that

$$p_{i,j}^{(k)}(\tau) = \begin{cases} \frac{\mu_1(k)}{\mu_0(k)+\mu_1(k)} \left(1 - e^{-[\mu_0(k)+\mu_1(k)]\tau}\right) & i = 1, j = 0, \\ \frac{\mu_0(k)}{\mu_0(k)+\mu_1(k)} \left(1 - e^{-[\mu_0(k)+\mu_1(k)]\tau}\right) & i = 0, j = 1, \\ \frac{\mu_0(k)+\mu_1(k)e^{-(\mu_0(k)+\mu_1(k))\tau}}{\mu_0(k)+\mu_1(k)} & i = 1, j = 1, \\ \frac{\mu_1(k)+\mu_0(k)e^{-(\mu_0(k)+\mu_1(k))\tau}}{\mu_0(k)+\mu_1(k)} & i = 0, j = 0 \end{cases} \tag{3.5}$$

for all $\tau > 0$.

We can further derive some properties for single path streaming from the continuous-time Gilbert model. The average packet loss rate for single path k is

$$P_k[\text{loss packet}] = \pi_1(k) = \frac{\mu_0(k)}{\mu_0(k) + \mu_1(k)}. \tag{3.6}$$

The error burst of length m for single path k is

$$\mathrm{P}_k[\text{error burst} = m] = \begin{cases} \pi_0(k)p_{0,1}^{(k)}(\delta_k)p_{1,0}^{(k)}(\delta_k) & \text{for } m = 1, \\ \pi_0(k)p_{0,1}^{(k)}(\delta_k)\left[p_{1,1}^{(k)}(\delta_k)\right]^{m-1} p_{1,0}^{(k)}(\delta_k) & \text{for } m \geq 2. \end{cases} \tag{3.7}$$

where δ_k is the packet spacing. Note that the probability of having error burst of m specifies the rate of occurence of error burst of m. Thus, the error burst of m would means that, at arbitary time, an error burst starts and the burst error is of length m.

The probability of having a packet error burst of any size is therefore

$$\mathrm{P}_k[\text{error burst}] = \sum_{m=1}^{\infty} \mathrm{P}_k[\text{error burst} = m] = \pi_0(k)p_{0,1}^{(k)}(\delta_k).$$

Moreover, the conditional probability of having a packet error burst of size $m \geq 1$, conditioned on there being a loss, is equal to

$$\begin{aligned} \mathrm{P}_{sp}[\text{error burst of size } m | \text{ error burst}] &= \frac{\mathrm{P}_k[\text{error burst} = m]}{\mathrm{P}_k[\text{error burst}]} \\ &= \left[p_{1,1}^{(k)}(\delta_k)\right]^{m-1} p_{1,0}^{(k)}(\delta_k) \quad \text{for } m \geq 1. \end{aligned} \tag{3.8}$$

In chapter 6, we will perform further analysis on the multi-path streaming with the continuous-time Gilbert model.

Chapter 4

Loss Recovery

The transmission of a packet over a best-effort network may not be reliable. Packets transmitted through the Internet will likely to go through packet-switched networks and being forwarded by the routers in the transmission path. The routers have buffers to store the received packets and forward the packets to other routers or hosts by looking up their stored route table with different policy (say First-come-first-serve). In case packets arrive to the router at a rate that exceed the routers routing capacity, the buffer will become full and the router will start to drop packets. This accounts for most of the packet loss experienced in end-to-end packet transmission on the Internet.

Two techniques, namely the Automatic Repeat Request (ARQ) and the Forward Error Correction (FEC) mechanisms, are commonly used to deal with the packet loss. Automatic Repeat Request (ARQ) schemes recover errors by retransmission of lost packets and Forward Error Correction (FEC) schemes recover errors by transmitting redundant information so that lost data can be recovered from the redundant information. The two techniques will be discussed in this chapter.

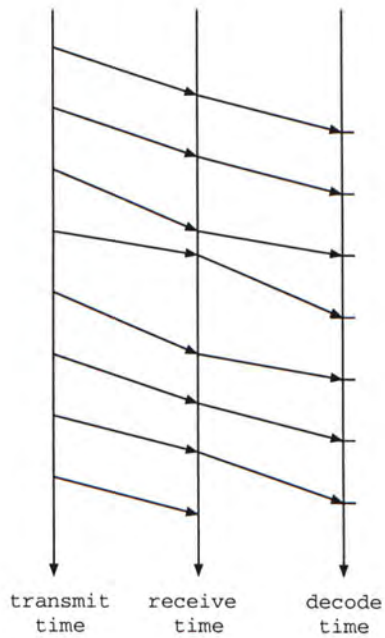


Figure 4.1: Transmission time and decode time

4.1 Automatic Repeat Request (ARQ)

Automatic Repeat Request (ARQ) schemes recover the loss by retransmission of lost packets. It is commonly used to implement reliable transmission on top of unreliable transmission (for example, in the TCP). The advantage of ARQ is that retransmission are required only in the case of packets loss. If there is no packet loss, no transmission overhead will be introduced to the network.

The main drawback of the ARQ is the high end-to-end latency. The loss can only be recovered when retransmitted packets arrive at the receiver and incurred the extra latency. Therefore, the total time elapsed before a retransmitted packet arrival is the sum of the receive time-out of the original packet, the transmission time for the retransmission request message transmitted from the receiver to the sender, and the transmission time for the retransmitted packet. The sum of all these times may be large on network with high latency network with high latency. However, streaming applications have a real-time constraint on delay.

Packets that arrive later than the decode-time deadline will usually be dropped or ignored. Figure 4.1 illustrates the the decode-time constraint of a constraint bit-rate streaming application. The packets are sent at a constant bit-rate from the server while the data are decoded at a constant bit-rate at the client. Retransmitted packets can not meet the decode-time deadline if the decode-time deadline are short. To have a long decode-time deadline, pre-buffering of considerable amount of data is required and we believe that ARQ is not suitable for streaming. Also, ARQ cannot take full advantages of using multiple paths as retransmitted packets are likely to go through the same path of the previous lost packet. If one of the paths have a high loss rate, the retransmitted packet will likely to be lost again without taking the advantage of the other better paths. We believe ARQ is good for error recovery for delay-insensitive application such as conventional file transfer but it is not suitable for streaming application. Hence, we do not consider using the ARQ for error recovery.

4.2 Forward Error Correction (FEC)

Forward Error Correction (FEC) Mechanisms recover loss by transmitting certain amount of redundant information together with original information. Loss of original information are recovered from the redundant information. Since redundant information is transmitted along with original information, no retransmission will be required for the reconstruction of lost data. Hence, FEC schemes do not experience the severe latency problem that ARQ schemes have. So FEC schemes become the primary choice for interactive applications that are delay-sensitive (say Internet phone) for error recovery. The tradeoff for the low latency is that redundant information is transmitted even there is no loss of data, which may be a waste of transmission bandwidth for a low loss rate path.

A naive implementation of forward error correction can be very inefficient. To illustrate, consider the forward error correction scheme that transmit a complete

duplicated copy along with the original data. This scheme will have 100% of redundant data and hence requires 100% more of bandwidth requirement. A simple analysis on this scheme will reveal that this error correction scheme has a poor performance. To simplify the analysis, assume that the probability of dropping a packet is p and the drop of each packet is independent. Since duplicated packets are transmitted, the data carried in a packet will be lost only if both the original and the duplicated packets are lost, which have probability of p^2. The probability of transmitting N data packets without any data loss is

$$(1 - p^2)^N$$

For instance, if N is 1000 and p is 0.02, the probability of no data loss will be $0.9996^{1000} = 0.670$. Thus, there is still a probability of 0.330 of having data loss, which is considered to be a poor performance.

It is therefore a crucial point to have efficient coding for the FEC We now study a FEC scheme with the error erasure codes implemented by Rizzo[16] which is very efficient in error correction. The error erasure codes scheme, or simply called erasure codes, consist of 2 components: The encoder and the decoder which perform the encoding and decoding of the erasure codes respectively.

The erasure codes encoder works in the following way: Given k data packets of same size (e.g., 1400 bytes in our implementation), l parity packets are constructed from the given k data packets. The total $n = k + l$ packets then forms a FEC group of size n. Among n packets in a FEC group, k of them carry exactly the original information and remaining $n - k$ parity packets carry redundant information. We call such code as the (n, k) code. The n packets of the FEC group are then transmitted to the destination. If k out of the n packets arrive to the destination, then the k packets carrying original information can be recovered without any information loss. Figure 4.2 illustrates the above flow. For example, if given $k = 80$ packets carrying the original information and 20 parity packets are constructed by the encoder to generate $(100, 80)$ codes, then this codes can

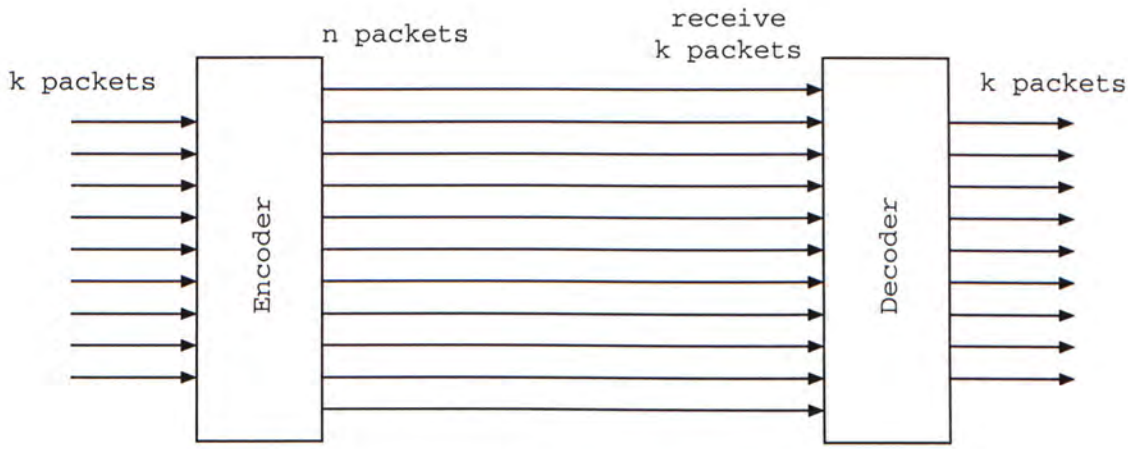


Figure 4.2: The Erasure Code Encoder and Decoder

tolerate up to 20 packets loss out of the 100 packets. The 100 packets are then transmitted to the destination. If 80 or more of them arrive (may these packets be the data packets or the redundant packets), the missing data packets can always be re-constructed and no information will be lost. However, if more than 20 packets are lost, say 25 of them are lost, then no recovery is possible. The information we lost depends on the number of the missing data packets. If the lost packets consist of 13 data packets and 12 parity packets we lose the information in the 13 data packets.

Let us consider a simple analysis on this coding scheme to show the efficiency of this erasure code, as compared to the naive approach previously shown. Again, we assume that the probability of dropping a packet is p and the drop of each packet is independent. Data loss occurs in a FEC group with (n, k) code if more than $n - k$ packets are dropped. For transmission of N data packets, the probability of having no data loss is

$$\left(\sum_{i=0}^{n-k} \binom{n}{i} p^i (1-p)^{n-k}\right)^{\lceil \frac{N}{k} \rceil}$$

We would use the same parameters as the previous example ($p = 0.02$, 100% redundant data) to compare the performance: for $p = 0.02$ and 100% redundancy

with fec parameters $k = 250$ and $n = 500$, the probability of transmission of 1000 data packets without any data loss will be $\left(\sum_{i=0}^{250} \binom{500}{i} 0.02^i 0.98^{500-i}\right)^4$ which yields a value very close to 1. This implies that the probability of having data loss will be less than 10^{-278}. Compared to the probability of having data loss using the naive approach (0.330), the performance of this erasure code have a much better performance in terms of the recovery capability. Thus, from this simple analysis, we can see that an efficient coding can dramatically increases the performance of the FEC scheme.

We briefly describe the working principle of the erasure code. The erasure code is based on the mathematical concepts in finite fields. Let $\vec{x} = x_0 \dots x_{k-1}$ be the data on the k different packets. We can generate the code $\vec{y} = y_0 \dots y_{n-1}$ from original data $\vec{x}$ by using a $n \times k$ generator matrix G by $\vec{y} = G\vec{x}$:

$$\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{k-1} \\ \vdots \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \dots \\ 0 & 0 & \dots & 0 & 1 \\ a_{0,0} & a_{0,1} & a_{0,2} & \dots & a_{0,k-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n-k-1,0} & a_{n-k-1,1} & a_{n-k-1,2} & \dots & a_{n-k-1,k-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \end{pmatrix}$$

or we can express as n equations:

$$\begin{aligned} y_i &= x_i && \text{for } i = 0, \dots, k-1 \\ y_i &= a_{i-k,0}x_0 + a_{i-k,1}x_1 + \dots + a_{i-k,k-1}x_{k-1} && \text{for } i = k, \dots, n-1 \end{aligned}$$

where $a_{i,j}$ are some constants. If any k subset of the above n equations are linearly independent, then we can determine x_i from the k subset equations. In other words, receiving any k components of $\vec{y}$ will be sufficient to determine $\vec{x}$. It is possible to have k subset of linearly independent equations by using $a_{i,j} = \alpha^{(i+1)j}$ for any prime α.

We now consider the issue of determining the amount of redundant information for the FEC. Using a small amount of redundant information helps in reducing the bandwidth overhead but it would also reduce the error recovery capability. On the other hand, using a large amount of redundant information helps to improve the error recovery capability but it would introduce more bandwidth overhead, which would be a waste if there are no or small losses.

For the (n, k) code, the choice of parameters n and k for the (n, k) code dictates the amount of redundant, as number of packets carrying redundant information is $n - k$. In order to determine the FEC group size, we have run some simulations to help determining the amount of redundant information under different network path condition. The result and analysis is shown in section 7.2.2 and the following conclusions are drawn:

- Increase amount of redundant information will increase the correcting capacity of the erasure codes.
- Even with fixed ratio $\frac{n}{k}$, increasing group size k can increasing the correcting capacity of the erasure codes, given a sufficient large ratio $\frac{n}{k}$.

The first conclusion is rather intuitive. By means of using codes with correcting capacity, we can recover more losses. The second conclusion is more interesting since even with same redundant information overhead (by fixing $\frac{n}{k}$), we may have different correcting capacity by using different FEC group size. Further analysis on this issue is shown in section 7.2.2.

Chapter 5

Connection Adaptation

The goal of connection adaptation is to adjust the connection of the paths among different servers that can hopefully maintaining the quality. We will discuss several related issues in this chapter. For example, we may have to determine which senders are the "best", which involve determining the path quality between the senders and the receiver. Another issue is that given multiple senders are used for the multi-path streaming, is it the best way for the senders to send equal share of data to the receiver? i.e. for n senders, each sender sends out $\frac{1}{n}$ data. In this chapter, we will consider related issues for making connection adaptations.

5.1 Path Quality

Estimating path quality is important to determine whether a path is good or not. Round-trip time is a common and simple quality measure as used in [1]. Another common technique is to use packet probes [17] to estimate path bandwidth. These measurements are based on the delay and the throughput of a path and may not well address the quality for streaming application. Path delay does not reflect any loss characteristics while throughput may not truly reflect the quality of a constant bit-rate streaming. A path with low throughput (given enough bandwidth) and low loss rate is preferred when compared to a path with high throughput but high loss rate. In contrast, we would like to understand

the loss characteristic of a path. A possible approach for the estimation is to determine the Gilbert model parameters μ_0 and μ_1 of each path. We can first determine the state transition probability $P_{01}^{(k)}$ and $P_{10}^{(k)}$ for each path k. The $P_{01}^{(k)}$ is the probability that a packet is lost given the previous packet is received. Thus, we can estimate $P_{01}^{(k)}$ statistically by counting number of the event packet i is received but packet $i+1$ is lost. Similarly, we can estimate $P_{10}^{(k)}$ by counting number of the event packet i is lost but packet $i+1$ is received. We can then determine $\mu_0(k)$ and $\mu_1(k)$ by using equation 3.5. First, we can obtain the ratio of $\mu_1(k)/\mu_0(k)$ by

$$\frac{\mu_1(k)}{\mu_0(k)} = \frac{P_{10}^{(k)}}{P_{01}^{(k)}}$$

Then we have

$$\begin{aligned} P_{0,1}^{(k)} &= \frac{\mu_0(k)}{\mu_0(k)+\mu_1(k)}\left(1 - e^{-[\mu_0(k)+\mu_1(k)]\tau}\right) \\ &= \frac{1}{(1+\alpha)}\left(1 - e^{-(1+\alpha)\mu_0(k)\tau}\right) \end{aligned}$$

We can then rewrite the equation as following to determine $\mu_0(k)$:

$$\mu_0(k) = -\frac{\ln[1-(1+\alpha)P_{0,1}^{(k)}]}{(1+\alpha)\tau} \tag{5.1}$$

where $\alpha = \frac{\mu_1(k)}{\mu_0(k)} = \frac{P_{10}^{(k)}}{P_{01}^{(k)}}$ and τ is the inter-packet spacing. We can also determine $\mu_1(k)$ by

$$\mu_1(k) = \alpha\mu_0(k) \tag{5.2}$$

The main drawback of this approach is that considerable number of samples is needed to make the result accurate. We will have further analysis on loss characteristics in chapter 6 and 7.

5.2 Effect of Shared Congestion Point

Congestion points of different paths may be shared or distinct. For paths that have distinct congestion points, we can model the dropping processes of the paths

with different independent Gilbert models. However, for paths that shared the same congestion point (i.e. packets from different paths are dropped at the same point), it will not be accurate to model the loss processes as different independent Gilbert models. A more appropriate approach is to model the loss processes of the paths under a single Gilbert model as the packets for 2 paths undergo the loss process at the shared single congestion point.

Shared point-of-congestion degrades the performance of multi-path streaming. Simulation result using ns-2 in section 7.3.8 shows that 2 paths with a common point-of-congestion results in performance similar to that of using a single path. Thus, shared point-of-congestion should always be avoided.

5.2.1 Point-of-Congestion Detection

Dan Rubenstein, Jim Kurose and Don Towsley have proposed a method to determine whether 2 paths shared the same point-of-congestion [18]. The working principle of the detection bases on the comparison of the loss correlation or the delay correlation of adjacent packets. If the adjacent packets are transmitted through the same congestion point then these adjacent packets should experience high loss or delay correlation. On the other hand, packets that are transmitted through different congestion points should experience no significant correlation on the loss or the delay. Therefore, it is possible to determine whether two paths share the same congestion point by measuring the loss or delay correlation for the cases of:

- adjacent packets from 2 different sources, and
- adjacent packets from one of the sources only.

For the point-of-congestion (POC) detection, we consider the Y-topology [18] only. Under Y-topology the senders for 2 flows are widely separated and the receivers of 2 flows are in the same local area network or machine. This is exactly

the same topology for the multiple-path streaming where a single receiver receives multiple streams (flows) from different distributed senders.

The measurement can be done either using loss correlation or delay correlation of the packets. For loss-correlation, the cross measure (M_x) and auto measure (M_a) and be obtained by:

$$M_x = \Pr(L_{2,i-1} = 0 | L_{1,j-1} = 0, adj_R(p_{1,j}, p_{2,i}) = 1)$$
$$M_a = \Pr(L_{2,i} = 0)$$

$L_{k,i} = 0$ when packet i from sender k is dropped before arriving the destination and $L_{k,i} = 1$ when packet i from sender k is received. And $adj_R(x, y) = 1$ when packet x and packet y are adjacent. The two paths between the receiver and the 2 senders share the same point-of-congestion if $M_x > M_a$.

The following example illustrate the calculation for loss-correlation point-of-congestion detection. Consider 20 **received** packets from sender 1 and sender 2:

$P_{1,1}$, $P_{1,2}$, $P_{2,1}$, $P_{2,3}$, $P_{1,4}$, $P_{1,7}$, $P_{2,4}$, $P_{2,5}$, $P_{2,8}$, $P_{1,9}$,
$P_{2,9}$, $P_{1,12}$, $P_{1,13}$, $P_{2,10}$, $P_{2,11}$, $P_{2,12}$, $P_{1,14}$, $P_{1,16}$, $P_{1,18}$, $P_{2,14}$.

For calculating cross measure M_x, the condition $L_{1,j-1} = 0$ and $adj_R(p_{1,j}, p_{2,i}) = 1$ is true for 3 cases:

- Case 1: $j = 7, i = 4$ ($P_{1,7}, P_{2,4}$)
- Case 2: $j = 9, i = 9$, ($P_{1,9}, P_{2,9}$), and
- Case 3: $j = 18, i = 14$ ($P_{1,18}, P_{2,14}$).

Among these 3 cases of adjacent packets, $L_{2,i-1} = 0$ is true for $i = 14$ (i.e. $P_{2,13}$ is lost but $P_{2,8}$ and $P_{2,3}$ are received). So the loss-corr cross measure M_x should be $\frac{1}{3}$. For calculating auto measure M_a, since 14 packets from sender 2 are expected ($P_{2,1}$ to $P_{2,14}$) but only 10 of them arrive ($P_{2,2}$, $P_{2,6}$, $P_{2,7}$, $P_{2,13}$ are lost), the loss-corr auto measure M_a should be $\frac{4}{14} = \frac{2}{7}$. In this example, $M_x > M_a$ so shared point-of-congestion is concluded.

5.3 Load Distribution

For multi-path streaming, data are streamed from multiple senders. One of the nature question to ask is that what are the best load distribution for each sender? i.e. Is there better way to stream data other than sending equal share from each sender? Intuitively, it may not be the best way to stream data with equal share from each sender. The reason is that different paths are likely to have different loss condition. If we model the loss processes of the paths with the Gilbert model, the parameter μ_0 and μ_1 for different paths are unlikely to be the same. It is a nature question to ask whether even or uneven distribution of load distribution give better performance.

We study the effect of using different load distribution among the paths using both the Gilbert model simulation and the ns-2 simulator in section 7.2.5 and 7.3.6 respectively. The results show that evenly distributing the load in simple round-robin manner gives fairly robust result when compared to uneven connection load distribution. Thus, evenly distribution of load in round-robin manner is preferred when compared to unevenly distribution because of the robustness and simplicity. Uneven distribution would require probe for path quality to determine the weight, which may require considerable amount of time and accuracy for the measurement of the path quality. Such probe is not required for round-robin load distribution.

Chapter 6

Analytical Evaluation

In this chapter, we will analyse the performance of the single-path and the multi-path streaming approaches. Our main focus is on loss characteristics. We first consider these approaches without the use of erasure codes, so as to understand the basic differences between single and multi-path streaming. We then also consider the changes in loss characteristics when an erasure code, and hence redundant information, is added, as this is another approach to dealing with packet losses. Specifically, we consider a variation of such codes, which we refer to as FEC, as defined in section 4.2. We use the Gilbert model, as our model of a path; as in [19] we characterize the path by its bottleneck link. This model, which is defined in chapter 3, allows for dependence in consecutive packet losses and should be a more accurate representation of the network than an independent loss model. The performance evaluation in this chapter is done at the receiving end of the paths.

We use the following performance measures to quantify the merits of the different streaming approaches (these are defined more formally below): section

1. mean data packets loss rate (with and without FEC),
2. conditional burst length distribution, conditioned on there being at least one error (with and without FEC),
3. lag-1 auto-correlation (with and without FEC).

The first performance measure is an obvious approach to comparing single and multi-path streaming (when losses, rather than throughput, are of importance). The other two performance measures are less obvious; however, we believe that they can significantly affect the quality of the viewed continuous media.

We will use the continuous time Gilbert model described in section 3.2.2 for the analysis.

Throughout the chapter we refer to single path streaming as SP streaming and multi-path streaming with N paths as MP streaming. Without loss of generality, when paths are homogeneous, we assume that SP streaming always transmits data along path 1. In the evaluation of MP streaming, we assume that the multiple paths have disjoint bottlenecks (or points of congestion) and hence the Gilbert models representing them are independent. Note that, since we represent a path by its bottleneck link, multiple paths with joint points of congestion could just be represented by a single Gilbert model. Lastly, note that our focus is on a streaming application which generates packets at a constant rate; hence our derivations below are done under this assumption.

6.1 Performance Analysis of SP vs. Multi-path Streaming (without FEC)

Let us first derive the average packet loss rate. Unless stated otherwise, below we consider a special case of multi-path streaming, namely dual path, round robin (DPRR) streaming. There are a number of different approaches to distributing data along the multiple paths; here we consider a simple case, i.e., DPRR, wherein each path carries half the application's traffic and the packet transmission is carried out in a round robin manner. That is, odd numbered packets are transmitted along path 1 while even numbered packets are transmitted along path 2. We use this simple scheme for dual path streaming to illustrate the basic

performance differences between single and multi-path streaming, so as to gain some basic understanding.

If we assume that the streaming rate does not affect the channel loss characteristics (i.e., the parameters of the Gilbert model), then for the SP case, the average packet loss rate is simply

$$P_{sp}[\text{loss packet}] = \pi_1(1) = \frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}. \tag{6.1}$$

For the MP case, assume that we have $N \geq 1$ paths and let α_i be the fraction of the application's workload that is sent along path i where $\sum_{i=1}^{N} \alpha_i = 1$. Then the average packet loss rate for the MP case is

$$P_{mp}[\text{loss packet}] = \sum_{i=1}^{N} \alpha_i \pi_1(i) = \sum_{i=1}^{N} \alpha_i \left(\frac{\mu_0(i)}{\mu_0(i)+\mu_1(i)} \right).$$

If these N paths are homogeneous, then we can simplify the above expression to

$$P_{mp}[\text{loss packet}] = \frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}. \tag{6.2}$$

Remark: the implication of Equations (6.1) and (6.2) is that if the application's sending rate does not affect the loss characteristics of the path then splitting the data between multiple homogeneous paths does *not* reduce the average packet loss rate, as compared to a single path with the same loss characteristics.

On the other hand, if the application's sending rate can affect the loss characteristics of the path (e.g., sending data with a higher bandwidth may increase the losses), then the average loss rate of the MP approach can be different from that of the SP approach. To illustrate this effect, let λ be the application's mean sending rate and

$$\mu_0(i) = \mathcal{F}(\lambda) \tag{6.3}$$

$$\mu_1(i) = \mathcal{B}(\lambda) \tag{6.4}$$

where $\mathcal{F}$ ($\mathcal{B}$) is a continuous non-decreasing (non-increasing) function of λ. Then, we have the following result.

Theorem 1 If the parameters of the Gilbert model are specified by functions $\mathcal{F}$ and $\mathcal{B}$, then the average packet loss rate under the single path streaming approach will be greater than or equal to the average packet loss rate under the multi-path streaming approach wherein these paths have the same Gilbert's parameters.

Proof: It is easy to show that the rate of change of the MP average packet loss rate under the homogeneous Gilbert model is:

$$\begin{aligned}\frac{dP_{mp}[\text{loss packet}]}{d\lambda} &= \frac{d}{d\lambda}\left[\frac{\mathcal{F}(\lambda)}{\mathcal{F}(\lambda)+\mathcal{B}(\lambda)}\right] \\ &= \frac{[\mathcal{F}(\lambda)+\mathcal{B}(\lambda)]\mathcal{F}'(\lambda)-\mathcal{F}(\lambda)[\mathcal{F}'(\lambda)+\mathcal{B}'(\lambda)]}{[\mathcal{F}(\lambda)+\mathcal{B}(\lambda)]^2} \\ &= \frac{\mathcal{B}(\lambda)\mathcal{F}'(\lambda)-\mathcal{F}(\lambda)\mathcal{B}'(\lambda)}{[\mathcal{F}(\lambda)+\mathcal{B}(\lambda)]^2} \geq 0.\end{aligned}$$

That is, a higher sending rate along a path results in a higher loss rate. Since the sending rate along a path in the MP case is less than or equal to the sending rate of the SP case, given that these paths are homogeneous, the resulting average packet loss rate of MP will be less than or equal to that of SP. ∎

Let us now consider the conditional burst length distribution, of both SP and MP cases, conditioned on there being a loss. Let λ_1 be the mean streaming rate (in units of packets per second) along path 1 and $\delta_1 = 1/\lambda_1$ is the time between two consecutively transmitted packets. Then, in the SP case (as also derived in [19] for a voice-over-IP type application), the probability of having a packet error burst of size $m \geq 1$ is:

$$\mathrm{P}_{sp}[\text{error burst} = m] = \begin{cases} \pi_0(1)p_{0,1}^{(1)}(\delta_1)p_{1,0}^{(1)}(\delta_1) & \text{for } m = 1, \\ \pi_0(1)p_{0,1}^{(1)}(\delta_1)\left[p_{1,1}^{(1)}(\delta_1)\right]^{m-1}p_{1,0}^{(1)}(\delta_1) & \text{for } m \geq 2. \end{cases} \tag{6.5}$$

The probability of having a packet error burst of any size is therefore

$$\mathrm{P}_{sp}[\text{error burst}] = \sum_{m=1}^{\infty}\mathrm{P}_{sp}[\text{error burst} = m] = \pi_0(1)p_{0,1}^{(1)}(\delta_1).$$

Moreover, the conditional probability of having a packet error burst of size $m \geq 1$, conditioned on there being a loss, is equal to

$$\begin{aligned} & \mathrm{P}_{sp}[\text{error burst of size } m | \text{ error burst}] \\ = & \frac{\mathrm{P}_{sp}[\text{error burst} = m]}{\mathrm{P}_{sp}[\text{error burst}]} \\ = & \left[p_{1,1}^{(1)}(\delta_1)\right]^{m-1} p_{1,0}^{(1)}(\delta_1) \qquad \text{for } m \geq 1. \end{aligned} \tag{6.6}$$

In the MP case, let us consider the special case of DPRR streaming, i.e., $N = 2$. Let λ_2 be the streaming rate (in units of packets per second) along path 1 or path 2. Note that under DPRR, $\lambda_2 = \lambda_1/2$. Then, the time between two consecutively transmitted packets along the same path is $\delta_2 = 1/\lambda_2 = 2\delta_1$. To understand the basic tradeoff between SP and MP streaming, we also assume that both paths are homogeneous such that they are characterized by a stationary continuous time Gilbert model of the same parameters (i.e., $\mu_0(1) = \mu_0(2)$ and $\mu_1(1) = \mu_1(2)$). Given this simplification, the stationary distributions for both paths are the same (i.e., $\pi_0(1) = \pi_0(2)$; $\pi_1(1) = \pi_1(2)$) and we can express all performance measures using the parameters of path 1. Under these assumptions, the probability of having a packet error burst of size $m \geq 1$ is:

$$\begin{aligned} & \mathrm{P}_{dp}[\text{error burst} = m] \\ = & \begin{cases} \pi_0(1)\pi_1(1)p_{0,0}^{(1)}(2\delta_1) & \text{for } m = 1, \\ \pi_0(1)\pi_1(1)\left[p_{1,1}^{(1)}(2\delta_1)\right]^{m-2} p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1) & \text{for } m \geq 2. \end{cases} \end{aligned} \tag{6.7}$$

and the probability of having a packet error burst of any size is therefore:

$$\begin{aligned} \mathrm{P}_{dp}[\text{error burst}] &= \sum_{m=1}^{\infty} \mathrm{P}_{dp}[\text{error burst} = m] \\ &= \pi_0(1)\pi_1(1)p_{0,0}^{(1)}(2\delta_1) + \sum_{m=2}^{\infty} \pi_0(1)\pi_1(1)[p_{1,1}^{(1)}(2\delta_1)]^{m-2} p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1) \\ &= \pi_0(1)\pi_1(1)\left[p_{0,0}^{(1)}(2\delta_1) + \frac{p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1)}{1 - p_{1,1}^{(1)}(2\delta_1)}\right] \\ &= \pi_0(1)\pi_1(1)\left[p_{0,0}^{(1)}(2\delta_1) + p_{0,1}^{(1)}(2\delta_1)\right] \\ &= \pi_0(1)\pi_1(1). \end{aligned}$$

Then, the conditional probability of having a packet error burst of size $m \geq 1$, conditioned on there being a packet error, is equal to:

$$\begin{aligned} & \mathrm{P}_{dp}[\text{error burst of size } m | \text{ error burst}] \\ = & \frac{\mathrm{P}_{dp}[\text{error burst} = m]}{\mathrm{P}_{dp}[\text{error burst}]} \\ = & \begin{cases} p_{0,0}^{(1)}(2\delta_1) & \text{for } m = 1, \\ \left[p_{1,1}^{(1)}(2\delta_1)\right]^{m-2} p_{0,1}^{(1)}(2\delta_1) p_{1,0}^{(1)}(2\delta_1) & \text{for } m \geq 2. \end{cases} \end{aligned} \tag{6.8}$$

We can now state the conditions under which the DPRR approach will have a small conditional burst error than the SP approach. Before we present this result, let us present the definition and a basic lemma of stochastic comparison [20].

Definition 1 We say that the random variable X is stochastically larger than the random variable Y, written $X \geq_{st} Y$, if $P[X \geq z] \geq P[Y \geq z]$ for all z.

Lemma 1 We say that $X \geq_{st} Y$ iff $E[f(X)] \geq E[f(Y)]$ for all increasing functions f.

Now, let $\mathcal{B}_{sp}$ and $\mathcal{B}_{dp}$ be the random variables representing the conditional packet error burst size, given that there is at least one packet error, under the SP and the homogeneous DPRR approaches, respectively. Then, we have the following result.

Theorem 2 If $p_{0,1}(2\delta_1)p_{1,0}(2\delta_1) \leq p_{1,1}(\delta_1)p_{1,0}(\delta_1)$, then $\mathcal{B}_{sp} \geq_{st} \mathcal{B}_{dp}$.

Proof: First, note that $p_{1,1}(t)$ is an non-increasing function of t. If $p_{0,1}(2\delta_1)p_{1,0}(2\delta_1) \leq p_{1,1}(\delta_1)p_{1,0}(\delta_1)$, then from Equations (6.6) and (6.8), we can deduce that

$\mathrm{P}_{dp}[\text{error burst of size } m | \text{ error burst}] \leq \mathrm{P}_{sp}[\text{error burst of size } m | \text{ error burst}] \quad m \geq 2.$

Since

$$\begin{aligned} \sum_{m=1}^{\infty} P_{sp}[\mathcal{B}_{sp} = m] &= \sum_{m=1}^{\infty} P_{dp}[\mathcal{B}_{dp} = m] = 1 && \text{and} \\ \sum_{m=j}^{\infty} P_{sp}[\mathcal{B}_{sp} = m] &\geq \sum_{m=j}^{\infty} P_{dp}[\mathcal{B}_{dp} = m] && \text{for } j \geq 2, \end{aligned}$$

we can conclude that $\mathcal{B}_{sp} \geq_{st} \mathcal{B}_{dp}$. ∎

Remark: Note that $\mathcal{B}_{sp} \geq_{st} \mathcal{B}_{dp}$ implies (based on Lemma 1) that $E[f(\mathcal{B}_{sp})] \geq E[f(\mathcal{B}_{dp})]$ for all increasing functions f. Therefore, we can conclude that for all moments of $\mathcal{B}_{sp}$ and $\mathcal{B}_{dp}$, we have $E[\mathcal{B}_{sp}^k] \geq E[\mathcal{B}_{dp}^k]$ for $k \geq 1$, where $E[\mathcal{B}_{sp}^k]$ and $E[\mathcal{B}_{dp}^k]$ refer to the k^{th} moments of $\mathcal{B}_{sp}$ and $\mathcal{B}_{dp}$, respectively. The implication of the above theorem is that the homogeneous DPRR approach will have a lower mean conditional burst length than the SP approach, given that the theorem's condition is satisfied.

Let us now consider the lag-1 autocorrelation of packet errors metric. We begin with the SP approach. The lag-1 autocorrelation function $R[X_t X_{t+\delta_1}]$ measures the degree of dependency of consecutive packet errors. For example, a high positive value of $R[X_t X_{t+\delta_1}]$ implies that a lost packet is very likely to be followed by another lost packet. On the other hand, a high negative value of $R[X_t X_{t+\delta_1}]$ implies that a lost packet is likely to be followed by a successful packet arrival. Also, if the statistics of the consecutive packet losses are not correlated[1] , then $R[X_t X_{t+\delta_1}] = 0$.

The lag-1 autocorrelation for the SP approach is

$$R[X_t X_{t+\delta_1}] = \frac{E[(X_t - \overline{X})(X_{t+\delta_1} - \overline{X})]}{E[(X_t - \overline{X})^2]} = \frac{E[X_t X_{t+\delta_1} - \overline{X}^2]}{E[X_t^2 - \overline{X}^2]}.$$

Since $\overline{X} = \pi_1(1) = \mu_0(1)/[\mu_0(1)+\mu_1(2)]$, $E[X_t X_{t+\delta_1}] = \pi_1(1) p_{1,1}^{(1)}(\delta_1)$ and $E[X_t^2] = \pi_1(1) = \mu_0(1)/[\mu_0(1)+\mu_1(2)]$, substituting these expressions into the above equation, gives us

$$R[X_t X_{t+\delta_1}] = \frac{\pi_1(1) p_{1,1}^{(1)}(\delta_1) - \pi_1^2(1)}{\pi_1(1)[1 - \pi_1(1)]} = \frac{[\mu_0(1) + \mu_1(1)] p_{1,1}^{(1)}(\delta_1) - \mu_0(1)}{\mu_1(1)}. \quad (6.9)$$

Lemma 2 For a high (low) bandwidth streaming application, the lag-1 autocorrelation of the SP streaming approach is positively correlated (tends to zero).

[1]Note that if the lag-1 autocorrelation, $R[X_t X_{t+\delta_1}]$, is equal to 0, it does not necessarily imply that consecutive packet losses are not correlated.

Proof: Note that when $\delta_1 \to 0$, $p_{1,1}^{(1)}(\delta_1) \to 1$, and consequently the lag-1 autocorrelation $R[X_t X_{t+\delta_1}]$ approaches 1. In other words, if the streaming application has a high bandwidth requirement such that the inter-packet spacing tends to zero, then the consecutive packet losses are *"positively"* correlated. On the other hand, when $\delta_1 \to \infty$, $p_{1,1}^{(1)}(\delta_1) \to \mu_0(1)/[\mu_0(1) + \mu_1(1)]$, and consequently the lag-1 autocorrelation $R[X_t X_{t+\delta_1}] \to 0$. This implies that for low bandwidth streaming applications, wherein the inter-packet spacing is very large, the lag1-autocorrelation tends to zero. ∎

Let us also derive the lag-1 autocorrelation of the homogeneous DPRR approach. The lag-1 autocorrelation in this case is:

$$R[X_t^{(1)} X_{t+\delta_1}^{(2)}] = \frac{E[(X_t^{(1)} - \overline{X^{(1)}})(X_{t+\delta_1}^{(2)} - \overline{X^{(2)}})]}{\sqrt{E[(X_t^{(1)} - \overline{X^{(1)}})^2] E[(X_t^{(2)} - \overline{X^{(2)}})^2]}}. \tag{6.10}$$

Because both paths are homogeneous (i.e., their respective Gilbert models have the same parameters), we can simplify the above expression as:

$$\begin{aligned} R[X_t^{(1)} X_{t+\delta_1}^{(2)}] &= \frac{E[X_t^{(1)} X_{t+\delta_1}^{(2)} - \overline{X^{(1)}}^2]}{E[(X_t^{(1)} - \overline{X^{(1)}})^2]} = \frac{E[X_t^{(1)} X_{t+\delta_1}^{(2)}] - E[\overline{X^{(1)}}^2]}{E[\left(X_t^{(1)}\right)^2] - E[\left(\overline{X^{(1)}}\right)^2]} \\ &= \frac{\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)\left(\frac{\mu_1(2)}{\mu_0(2)+\mu_1(2)}\right) - \left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)^2}{\mu_0(1)\mu_1(1)/\left(\mu_0(1)+\mu_1(1)\right)^2} \\ &= \frac{\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)^2 - \left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)^2}{\mu_0(1)\mu_1(1)/\left(\mu_0(1)+\mu_1(1)\right)^2} \\ &= 0 \end{aligned} \tag{6.11}$$

In fact, we can see that the consecutive packet losses under the homogeneous DPRR application are *"uncorrelated"* since we have assumed independence of the two paths.

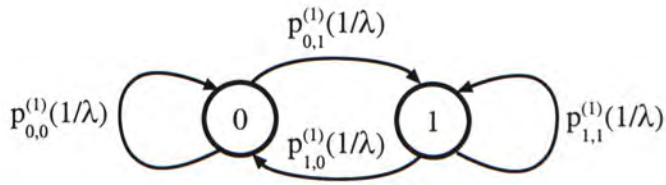


Figure 6.1: An Embedded Markov Chain which describes whether a transmitted packet is loss or not.

6.2 Performance Analysis of SP vs. Multi-path Streaming (with FEC)

We have shown that loss characteristics can be improved with multi-path streaming as compared to single path streaming, under conditions and metrics specified above. However, an interesting question that remains is whether there are still benefits to be gained once some form of redundancy is added to the stream. Specifically, we consider the use of an erasure code (as defined below), to which we will refer as FEC in the remainder of the paper. Hence, in this section we focus on the basic understanding of the performance of single path vs. multi-path streaming when FEC is added to the stream.

Since numerous coding schemes exist, we first give the details of the simple FEC scheme considered here. As discussed in section 4.2, we divide a video file into groups of data packets such that each group consists of k data packets. Given each group of k data packets, we generate $n > k$ packets. We refer to these n packets as a FEC group. The encoding scheme is such that, if the number of lost packets within a FEC group is less than or equal to $(n - k)$, then we can reconstruct the original k data packets within that FEC group.

Let us first derive the average packet loss rate under the SP approach. As before, assume that we use path 1 which is characterized by a Gilbert model, as defined above, with parameters $\mu_0(1)$ and $\mu_1(1)$. The streaming application generates packets at a rate of λ (in unit of packet/sec)[2] . Whenever a packet is

[2]Note that here, "packets" includes both data packets and packets carrying redundant

transmitted along this path, it may be lost (if the state of the path is "1") or it may arrive successfully at the receiver (if the state of the path is "0"). Figure 6.1 depicts an embedded Markov chain of this path wherein the two consecutive embedded points are $1/\lambda$ units apart. The derivation of transition probabilities of this DTMC is based on Equation (3.5); hence they are a function of the Gilbert model's parameters $\mu_0(1)$ and $\mu_1(1)$ as well as the packet transmission rate λ. The steady state probabilities of this embedded Markov chain are $\pi_0(1) = \frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}$ and $\pi_1(1) = \frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}$.

We are now interested in deriving $P^{(1)}(j,n)$, which is the probability of losing j packet in an n packet transmission. We define

$$P_i^{(1)}(j,n) = \text{Prob}(j,n|\text{initial state of the path is } i) \qquad i \in \{0,1\}$$

as the probability of j lost packet in an n packet transmission, given that the *first* packet was transmitted when the path was in state i (where $i \in \{0,1\}$). We then have:

$$P^{(1)}(j,n) = P_0^{(1)}(j,n)\pi_0(1) + P_1^{(1)}(j,n)\pi_1(1) \qquad j = 0,1,\ldots,n. \quad (6.12)$$

We also define:

$$L_i^{(1)}(j,n) = \text{Prob}(j,n|\text{the initial state of the path is } i \text{ and the final state is } 0) \qquad i \in \{0,1\}$$

$$H_i^{(1)}(j,n) = \text{Prob}(j,n|\text{the initial state of the path is } i \text{ and the final state is } 1) \qquad i \in \{0,1\}$$

where $L_i^{(1)}(j,n)$ ($H_i^{(1)}(j,n)$) is the probability that we have j lost packets in an n packet transmission, given that the *first* packet was transmitted when the path was in state i (where $i \in \{0,1\}$) and that the *last* packet was transmitted when the path was in state 0 (state 1). Then we have:

$$P_i^{(1)}(j,n) = L_i^{(1)}(j,n) + H_i^{(1)}(j,n) \qquad i \in \{0,1\},\ j = 0,1,\ldots,n. \quad (6.13)$$

information.

We can also express $L_i^{(1)}(j,n)$ and $H_i^{(1)}(j,n)$ in the following recursive forms:

$$L_i^{(1)}(j,n) = L_i^{(1)}(j,n-1)(1-p_{0,1}^{(1)}(1/\lambda)) + H_i^{(1)}(j,n-1)p_{1,0}^{(1)}(1/\lambda) \quad \text{for } j < n, \quad (6.14)$$

$$H_i^{(1)}(j,n) = L_i^{(1)}(j-1,n-1)p_{0,1}^{(1)}(1/\lambda) + H_i^{(1)}(j-1,n-1)(1-p_{1,0}^{(1)}(1/\lambda)) \quad \text{for } j < n. \quad (6.15)$$

where we also have the following boundary conditions:

$$L_i^{(1)}(j,m) = 0 \qquad i \in \{0,1\};\ j = 0,\ldots,n;\ m \le j \quad (6.16)$$

$$L_0^{(1)}(0,m) = (1-p_{0,1}^{(1)}(1/\lambda))^{m-1} \quad \text{for } m = 1,\ldots,n \quad (6.17)$$

$$L_1^{(1)}(0,m) = 0 \quad \text{for } m = 1,\ldots,n \quad (6.18)$$

$$H_i^{(1)}(j,m) = 0 \qquad i \in \{0,1\};\ j = 1,\ldots,n;\ m < j \quad (6.19)$$

$$H_i^{(1)}(0,m) = 0 \quad \text{for } i \in \{0,1\};\ m = 0,\ldots,n \quad (6.20)$$

$$H_0^{(1)}(m,m) = 0 \quad \text{for } m = 1,\ldots,n \quad (6.21)$$

$$H_1^{(1)}(m,m) = (1-p_{1,0}^{(1)}(1/\lambda))^{m-1} \quad \text{for } m = 1,\ldots,n. \quad (6.22)$$

Remark: To compute the value of $P^{(1)}(j,n)$ in Equation (6.12), we need to compute the values of the four square matrices $\boldsymbol{L}_0^{(1)}, \boldsymbol{L}_1^{(1)}, \boldsymbol{H}_0^{(1)}$, and $\boldsymbol{H}_1^{(1)}$, whose entries can be computed using Equations (6.14) through (6.22). Each of these matrices is of size $(n+1)\times(n+1)$. In other words, computing the values of $P^{(1)}(j,n)$ (for all j) has a computational complexity of $\Theta(4(n+1)^2)$.

Let P_{sp} be the probability of an irrecoverable error within a FEC group. It is equal to

$$P_{sp} = \sum_{j=n-k+1}^{n} P^{(1)}(j,n) = \sum_{j=n-k+1}^{n} \left[P_0^{(1)}(j,n)\pi_0(1) + P_1^{(1)}(j,n)\pi_1(1)\right]$$
$$= \sum_{j=n-k+1}^{n} \left[\left(L_0^{(1)}(j,n) + H_0^{(1)}(j,n)\right)\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right) + \right.$$
$$\left. + \left(L_1^{(1)}(j,n) + H_1^{(1)}(j,n)\right)\left(\frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}\right)\right].$$

To derive the average data packet loss rate (with use of FEC) for the SP approach, denoted by $\mathcal{L}_{sp}$, we consider the following two cases, based on the number of lost packets, $j \in \{0, 1, \ldots, n\}$, within a FEC group.

Case 1: $j \leq n - k$

If j, the number of lost packet within a FEC group, is less than or equal to $n - k$, then all k data packets can be reconstructed at the receiver. Hence, this case does not contribute to *information* loss and $\mathcal{L}_{sp} = 0$.

Case 2: $j > n - k$

In this case, the lost data packets cannot be fully reconstructed and some information will be lost. However, given that there j lost packets within a FEC group, there are a number of different ways to distribute these losses among the n packets of the FEC group. To understand this effect, let us illustrate it using an example. Assume that $n = 5$ and $k = 4$. If $j = 2$, then there are two possible ways to distribute these two lost packets among the packets of the FEC group: (1) the two lost packets are the data packets within the FEC group, or (2) one lost packet is a data packet and the other lost packet corresponds to redundant information in the FEC code. In the first case, we lost 2 data packets out of a 4 data packet transmission. In the second case, we lost 1 data packet out of a 4 data packet transmission. Using the same argument, if $j = 5$, then there is only one way to distribute these five lost packets among packets of the FEC group. That is, all data packets are lost. Therefore, given that there are j lost packets, the number of ways to distribute the j lost packets among the packets of a FEC group is $\mathcal{W} = \mathcal{M} - j + (n - k) + 1$ where $\mathcal{M} = \min\{j, k\}$. Let $\mathcal{L}(j)$ be the average data packet loss rate given that there are j lost packets in a FEC group. Then,

we have

$$\begin{aligned}\mathcal{L}(j) &= \frac{1}{\mathcal{W}} \sum_{i=j-(n-k)}^{\mathcal{M}} \frac{i}{k} \\ &= \left(\frac{1}{\mathcal{M}-j+(n-k)+1}\right)\left(\frac{1}{k}\right) \\ &\quad \times \left(\frac{\mathcal{M}(\mathcal{M}+1)}{2} - \frac{(j-(n-k))(j-(n-k)-1)}{2}\right) \quad (6.23)\end{aligned}$$

It is now easy to derive $\mathcal{L}_{sp}$, the average data packet loss rate (with the use of FEC) for the SP approach as follows:

$$\begin{aligned}\mathcal{L}_{sp} &= \sum_{j=n-k+1}^{n} P^{(1)}(j,n)\mathcal{L}(j) \\ &= \sum_{j=n-k+1}^{n} \left[P_0^{(1)}(j,n)\pi_0(1) + P_1^{(1)}(j,n)\pi_1(1)\right]\mathcal{L}(j) \\ &= \sum_{j=n-k+1}^{n} \left[\left(L_0(j,n)+H_0(j,n)\right)\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)\mathcal{L}(j) + \right. \\ &\qquad \left. \left(L_1^{(1)}(j,n)+H_1^{(1)}(j,n)\right)\left(\frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}\right)\mathcal{L}(j)\right]. \quad (6.24)\end{aligned}$$

To derive the average data packet loss rate (with use of FEC) for the MP approach, let us first consider a simple case of dual-path streaming. Assume that there are two servers S_1 and S_2 that use two different, possibly heterogeneous, paths. We use the same FEC scheme as described above to generate a stream of data divided into n packet FEC groups. To transmit the packets within a FEC group, server S_1 transmits n_1 packets while server S_2 transmits n_2 packets such that $n_1 + n_2 = n$. Based on the similar argument we made above in the SP case, we have

$$P^{(1)}(j,n_1) = P_0^{(1)}(j,n_1)\pi_0(1) + P_1^{(1)}(j,n_1)\pi_1(1) \qquad j = 0,1,\ldots,n_1 \quad (6.25)$$

$$P^{(2)}(j,n_2) = P_0^{(2)}(j,n_2)\pi_0(2) + P_1^{(2)}(j,n_2)\pi_1(2) \qquad j = 0,1,\ldots,n_2. \quad (6.26)$$

The computation of $P_i^{(h)}(j,n_h)$ where $i \in \{0,1\}$ and $h \in \{1,2\}$ is similar to the approach mentioned above, that is, by evaluating the entries of the corresponding

four matrices. The computational complexity would then be $\Theta(4(n_1+1)^2+4(n_2+1)^2)$.

Let P_{2p} be the probability of an irrecoverable error within a FEC group. It is equal to

$$P_{2p} = \sum_{j=n-k+1}^{n} \sum_{h=0}^{j} P^{(1)}(h, n_1) P^{(2)}(j-h, n_2), \tag{6.27}$$

which involves a convolution between the two probability mass functions, $P^{(1)}(j, n_1)$ and $P^{(2)}(j, n_2)$. Let $\mathcal{L}_{2p}$ be the average data packet loss rate (with use of FEC) for the dual path approach. Then, we have

$$\mathcal{L}_{2p} = \sum_{j=n-k+1}^{n} \sum_{h=0}^{j} P^{(1)}(h, n_1) P^{(2)}(j-h, n_2) \mathcal{L}(j). \tag{6.28}$$

In general, if we employ N servers $S_1, S_2, \ldots, S_N$, then the probability of an irrecoverable error within a FEC group is

$$P_{Np} = \sum_{j=n-k+1}^{n} \left(\sum_{i_1+\ldots+i_N=j} P^{(1)}(i_1, n_1) P^{(2)}(i_2, n_2) \cdots P^{(N)}(i_N, n_1) \right). \tag{6.29}$$

The average data packet loss rate with FEC under a MP streaming with N paths is

$$\mathcal{L}_{Np} = \sum_{j=n-k+1}^{n} \left(\sum_{i_1+\ldots+i_N=j} P^{(1)}(i_1, n_1) P^{(2)}(i_2, n_2) \cdots P^{(N)}(i_N, n_1) \right) \mathcal{L}(j). \tag{6.30}$$

In the case of the other two performance measures, namely, the conditional burst length distribution and the lag-1 autocorrelation, we resort to the use of simulation, as described in the following chapter.

Chapter 7

Experiments and Simulations

7.1 Effect of Correlated Bursty Losses on Video Quality

In this experiment, we drop 2% of the frames from video $\mathcal{V}$. These 2% losses are introduced in a variety of "patterns", e.g., the dropped frames can be evenly spaced throughout video $\mathcal{V}$, or they can be more bursty. The details of which frames are dropped, given a particular drop pattern as identified by the burst length, are given in the first two columns of Table 7.1. Moreover, in evaluating the quality of the resulting video $\mathcal{V}$, we use a common error concealment scheme to make up for a dropped frame. Specifically, a dropped frame is replaced by the previous frame which is successfully received. For example, frame i replaces frames $i+1, i+2, \cdots, i+k$ if frame i is received successfully and frames $i+1, \cdots, i+k$ are lost.

For each possible frame loss pattern, we measure the quality of the received video by computing the peak signal-to-noise ratio (PSNR) as follows. (Note that, a larger value of PSNR implies a higher quality of the video.) In general, for a video of l frames where each frame consists of $m \times n$ pixels, (each containing an RGB value[1] with each of the three colors represented by 8-bits), the PSNR is

[1]Information about the three colors, red, green, and blue.

Error Burst Length	Lost Frames Numbers	PSNR (dB)
1	25+k*50 where k$\in \{0, 1, \cdots, 29\}$	39.107 dB
2	{50,51} + k*100 where k$\in \{0, 1, \cdots, 14\}$	38.015 dB
3	{74,75,76} +k*150 where k$\in \{0, 1, \cdots, 9\}$	31.325 dB
5	{123,124,125,126,127} +k*200 where k$\in \{0, 1, \cdots, 5\}$	30.433 dB
15	{368,369,...,381,382} +k*750 where k$\in \{0, 1\}$	28.407 dB
30	{736,737,...,764,765}	29.942 dB

Table 7.1: Peak signal-to-noise ratio (PSNR) for various bursty loss patterns.

calculated using the following expression (in dB):

$$SNR_{peak} = 10 \times \log_{10} \frac{255^2}{\left(\frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{l} \sum_{c=1}^{3} (P_1(i,j,k,c) - P_2(i,j,k,c))^2}{3 \times m \times n \times l} \right)}.$$

where $P_s(i, j, k, c)$ is the pixel value at coordinate (i, j) of k-th video frame (of stream s, $s = 1, 2$) and color channel c where $c = 1, 2, 3$, for red, green, and blue, respectively. In our experiment, the values of m,n, and l are 352, 240 and 1500, respectively. The source video in this experiment is using MPEG-1 NTSC settings [21] where each frame is 352×240 (with 29.97 frames per second), hence the values of m and n above. Also, we use approximately the first 50 seconds of this video for this experiment, hence the value of l above. Values for P_1 are obtained from the frame sequence resulting after the drop-and-conceal process while values for P_2 are obtained from the original video frames of $\mathcal{V}$. Table 7.1 gives the PSNR values for the different burst patterns. We can observe that given the same amount of information loss (e.g., 2% in our experiment), the PSNR metric can be significantly lower for the more bursty loss patterns, and hence is the quality of the video. Thus, we believe that burst length distribution and correlations between losses are the right metrics for evaluating the goodness of a streaming approach as they directly reflect on the quality of received video.

7.2 Analytical Model Based Evaluation

In this section, we further evaluate the loss characteristics of the SP vs. MP methods using simulations of the Gilbert model described in chapter 6. The simulations allow us to consider the loss characteristics under more sophisticated scenarios than in chapter 6. Specifically, we assume an MPEG-1 video streaming application which generates packets at a rate of 120 packets per second with each packet containing 1400 bytes. We consider at most three senders (S_1, S_2, S_3) and one receiver C. Sender S_i uses path i to transmit its fraction of the data; unless otherwise stated, these paths are assumed to be independent. Moreover, in the figures given below (unless otherwise stated), the curves corresponding to SP streaming use path 1, the curves corresponding to MP streaming with 2 senders use paths 1 and 2, and the curves corresponding to MP streaming with 3 senders use all three paths. Unless stated otherwise, the packet assignment is carried out in a round-robin manner, e.g., if we use all three senders, then sender S_i transmits data packets at a rate of 40 packets per second. The loss process of path i is modeled by a continuous stationary Gilbert model (as defined in chapter 6). Unless stated otherwise, we use $\mu_0(i) = 20$ and $\mu_1(i) = 70$, for $i = 1, 2, 3$. Lastly, we consider all the same performance metrics as defined in chapter 6.

7.2.1 Data Loss Rate

In this experiment, we study the data packet loss rate of the SP and MP approaches, using only two paths, 1 and 2. The path parameters are as described above except that we vary the $\mu_0(2)$ parameter from 5 to 50. Table 7.2 illustrates the data loss rate for the single path(s) and the dual-path approaches (in each case, with and without the use of FEC, where the parameters for the FEC scheme are $n = 5$ and $k = 4$). We can observe that in this experiment:

- Without the use of FEC, the data packet loss rate of the dual path is approximately the mean of the data packet loss rates of paths 1 and 2.

These results are consistent with the derivation in chapter 6.

- With the use of FEC, (in this case $n = 5$ and $k = 4$), the achieved data packet loss rate can be *less* than the average of the data packet loss rates of the two corresponding single paths. This may occur due to the fact that error burst lengths in dual-path streaming tend to be shorter than in single-path streaming (refer Theorem 2 in chapter 6), and hence a chance of recovery of lost data (using FEC) should also be higher.

This experiment also illustrates the potential advantages of multi-path streaming over "best path" streaming, even when losses (rather than throughput) are the important consideration. That is, when multiple paths are available (but throughput is not the issue), another approach might be to stream the data over the "best" available path (and as congestion conditions change keep switching the streaming of the data to the best available path at the time). Our experiment shows that MP streaming could provide better loss characteristics (e.g., when FEC is used) than the "best" available path. (Please refer to experiment in section 7.2.6 below on further comparison to a best-path type approach.)

Loss rate: ($\mu_0(2)$)	single path: path 1 w/o FEC	single path: path 2 w/o FEC	dual-path without FEC	single path: path 1 with FEC	single path: path 2 with FEC	dual-path with FEC
5	0.221743	0.066767	0.144351	0.189053	0.053048	0.101264
15	0.221743	0.176153	0.199395	0.189053	0.147171	0.141632
20	0.221743	0.221743	0.222255	0.189053	0.189053	0.158861
35	0.221743	0.332848	0.278178	0.189053	0.297647	0.201947
50	0.221743	0.416609	0.319230	0.189053	0.385602	0.235681

Table 7.2: Data Loss rate with Heterogeneous Paths.

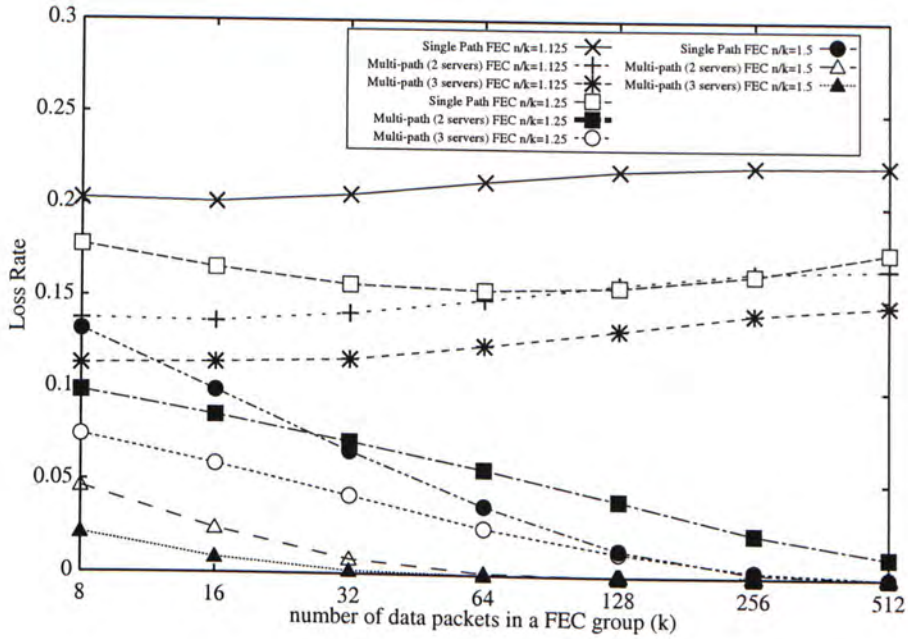


Figure 7.1: Loss rate as a function of n/k and k

7.2.2 Data Loss Rate as a function of FEC parameters

In this experiment, we study the effects of FEC parameters on the data loss rate. In general, there are two ways to vary the FEC parameters. We can:

1. Increase the degree of redundancy (e.g., for a given value of k, increase the value of n). Note that by increasing the degree of redundancy, we also increase the amount of traffic on the network.

2. Increase the values of n and k but keep the same ratio of n/k. This implies that we increase the FEC group size, and hence the application needs to maintain a larger receiving buffer (for reconstruction purposes in case of loss) as well as experience potentially higher latency (since a larger amount of information must be received prior to reconstruction of missing information).

Figure 7.1 illustrates the effects of FEC parameters on the data loss rate, and specifically, it depicts data loss rates for SP and MP streaming with $n/k = 1.125, 1.25$ and 1.5 as well as with different FEC group sizes (where we vary the

number of data packets in a FEC group (k) from 8 to 512 packets). In this case the path parameters are $\mu_0(1) = 20$, $\mu_1(1) = 70$, $\mu_0(2) = \mu_0(3) = 10$, and $\mu_1(2) = \mu_1(3) = 80$. We observe that:

- Increasing the amount of redundancy (e.g., from $n/k = 1.125$ to 1.5) in SP or MP streaming can reduce the data loss rate. However, one can achieve a lower data packet loss rate with MP streaming with a smaller n/k ratio (as compared to SP streaming). In other words, without introducing additional network traffic, we can obtain better performance with MP streaming.

- Increasing the number of data packets in a FEC group (while keeping the same ratio of n/k) may not necessary reduce the data loss rate. For example, consider SP streaming; as we increase k, the data loss rate actually increases in some cases. The maybe explained by a possible "convergence" of the data loss rate, as a function of n and k, to a non-zero value (please refer to the explanation followed).

We provide an explanation for the possible convergence of the data loss rate when the FEC group size is increased (e.g., by keeping the ratio of n/k but increasing the value of n).

Let $P_{p-path}(j,n)$ be the probability of losing j packets under p parallel senders/paths when the FEC group size is n. Based on the derivation in chapter 6, we have:

$$\begin{aligned} P_{1-path}(j,n) &= P^{(1)}(j,n) \qquad \text{for } j = 1,2,\ldots,n. \\ P_{m-path}(j,n) &= \sum_{i_1+\cdots+i_m=j} P^{(1)}(i_1,n_1) * P^{(2)}(i_2,n_2) * \cdots * P^{(m)}(i_m,n_m) \\ &\qquad \text{for } n_1+\cdots+n_m = n \text{ and } m > 1. \end{aligned}$$

Let $\Psi_{N-path}(n)$ be the average number of lost packets when we use $N \geq 1$ parallel senders and the FEC group size is n. We have that

$$\Psi_{N-path}(n) = \sum_{j=n-k+1}^{n} j P_{N-path}(j,n).$$

Let $\sigma = \frac{n-k}{n}$ be the fraction of redundant packets within a FEC group and P_{N-path} be the probability of losing any packet when one uses N parallel senders. We have that $P_{N-path} = \sum_{i=1}^{N} \alpha_i \frac{\mu_0(i)}{\mu_0(i)+\mu_1(i)}$. Let $\mathcal{L}_{p-path}(n,\sigma)$ denote the average data loss rate when a FEC group size of size n is used and σn is the number of redundant packets with $p \geq 1$ parallel senders. We conjecture that

$$\lim_{n\to\infty} \mathcal{L}_{p-path}(n,\sigma) = \begin{cases} 0 & \text{if } \lim_{n\to\infty} \frac{\Psi_{N-path}(n)}{\sigma n} \to 0, \\ (0, P_{N-path}] & \text{otherwise.} \end{cases} \quad (7.1)$$

The above statement is intuitive for the following reasons. As we increase n (but keep σ constant), if the rate of increase of $\Psi_{N-path}(n)$ is less than the rate of increase of σn, then we will have more redundant packets to "protect" the lost packets within a FEC group; in that case, the average data loss rate $\mathcal{L}_{p-path}(n,\sigma)$ will converge to zero as we increase n. On the other hand, if the rate of increase of $\Psi_{N-path}(n)$ is greater than the rate of increase of σn, as we increase n, then we will have some irrecoverable packet losses within a FEC group. In that case, $\mathcal{L}_{N-path}(n,\sigma)$ has to be greater than zero and in the *worst* case, it is upper bounded by the packet loss rate of the channel.

7.2.3 Conditional Error Burst Length

In this experiment, we compare the conditional burst length distribution, conditioned on there being at least one error. Figure 7.2 illustrates the conditional probability mass functions of error burst length (as defined in chapter 6). In this experiment, we observe that the packet error burst length is indeed stochastically less than the error burst length of the single path streaming. We also note, that the condition of Theorem 2 in chapter 6 holds in this experiment[2] . This relationship also holds when we employ FEC.

[2]Note that here we illustrate the probability mass function rather than the probability distribution function, as we believe it depicts the results of the experiment better.

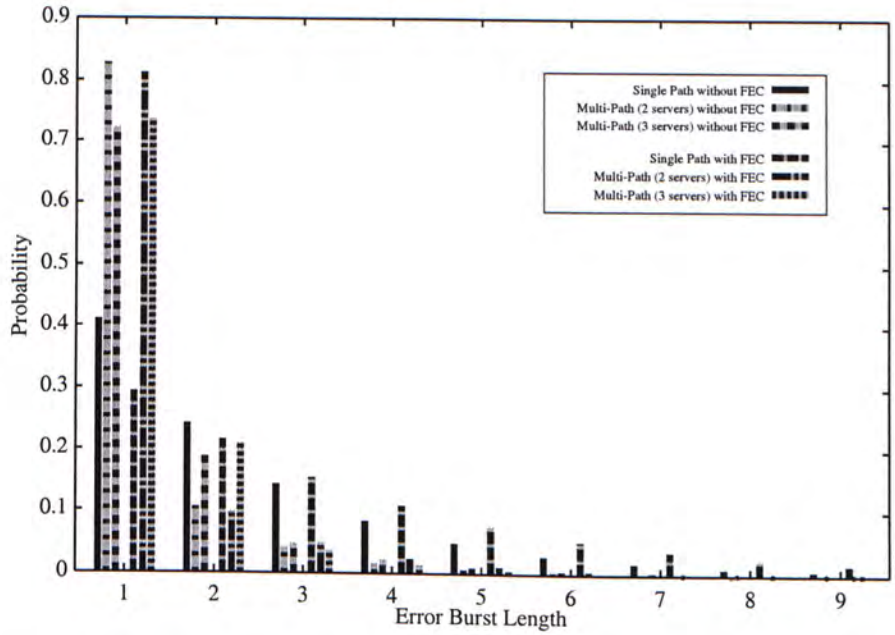


Figure 7.2: Conditional probability mass functions of error burst length.

7.2.4 Lag-1 Autocorrelation

In this experiment, we study the lag-1 autocorrelation of packet losses for both SP and MP streaming (as defined in chapter 6). Figure 7.3 illustrates the lag-1 autocorrelation where $\mu_1(1) = \mu_1(2) = \mu_1(3) = 70$ and $\mu_0(i)$ is varied (identically) for all three paths. We make the following observations.

- When we use MP streaming without FEC, the lag-1 autocorrelation is nearly zero while the lag-1 autocorrelation of SP path streaming (with or without FEC) can be highly correlated.

- The use of FEC may increase the lag-1 autocorrelation (for both SP and MP approaches). This may be explained as follows. The irrecoverable losses (after the error correction process) are likely to end up "closer" in the resulting data stream than in the original data stream (one without the use of erasure codes), and hence the lag-1 autocorrelation in this new stream behaves similarly to lag-h autocorrelation of the original stream, where $h > 1$. However, we still observe that the lag-1 autocorrelation of

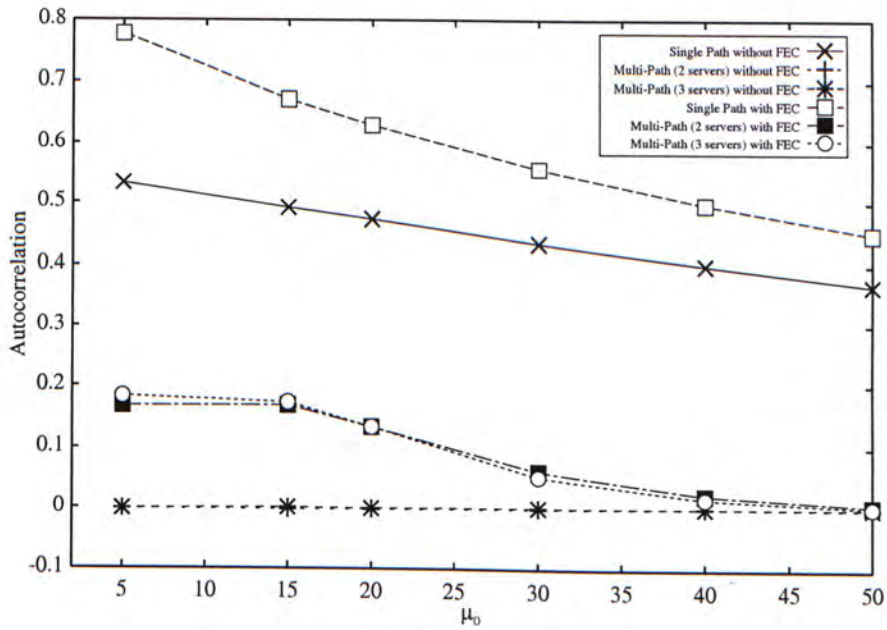
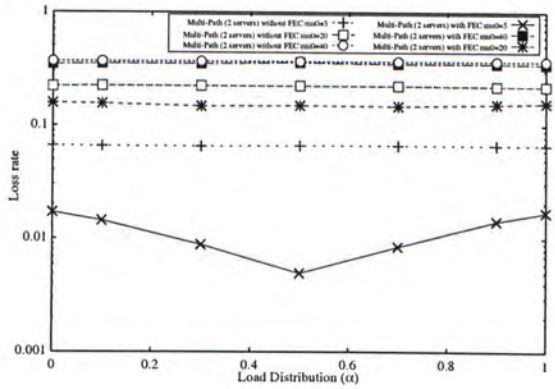


Figure 7.3: Lag-1 autocorrelation.

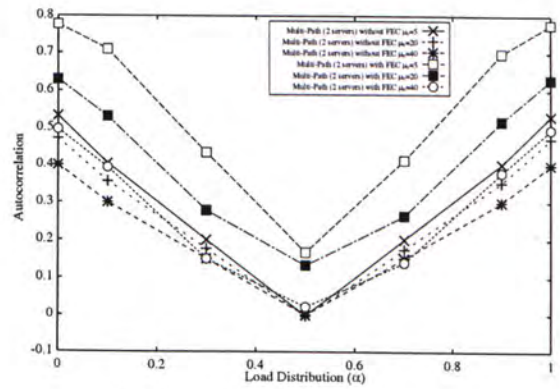
MP streaming is significantly closer to zero as compared to SP streaming, even with the use of FEC.

7.2.5 Effects of Load Distribution Among Senders

In previous experiments, all senders transmitted packets in a round-robin manner and hence the load distribution between all the senders was the same. In this experiment, we investigate effects of load distribution among senders. Specifically, we distribute the load among two senders only, where parameter α refers to the fraction of packets sent by sender 1. For instance, when $\alpha = 0.3$, sender 1 sends 30% of the packets while sender 2 sends 70% of packets. In the cases of $\alpha = 0$ and $\alpha = 1$, this degenerates to single path streaming using path 1 and path 2, respectively. Both path 1 and path 2 have the same parameters with $\mu_0 = 5, 20$, or 40 and μ_1 fixed at 70. Figure 7.4 illustrates results of this experiment. We observe that there is a slight improvement in loss rate when FEC is used and the load is equally distributed between the two senders. Moreover, in this experiment, the



(a) Loss Rate



(*b*)Lag-1 Autocorrelation

Figure 7.4: Loss rate and Lag-1 autocorrelation for different load distributions for the dual-path streaming

lag-1 autocorrelation reaches its minimum value under equal load distribution. This implies that simple round-robin packet distribution among paths should result in a higher quality of received video. That is, this simple approach of equal distribution is fairly robust.

7.2.6 Sensitivity Analysis

In this experiment, we study the relative performance of MP streaming vs. SP streaming when the SP streaming is performed over the *best* of the available paths. For example, if the performance metric is loss rate, then the path with the lowest loss rate is used. We note that implementation of this form of *best* single path streaming would likely require a fairly accurate monitoring of the loss characteristics of a path; otherwise, the wrong path might be selected. That is, the sensitivity (or robustness) of the streaming decisions to the accuracy of the available information about the network is an important issue.

In this sensitivity experiment, we consider a two-path system, where the fixed parameters are $\mu_0(1) = 20$ and $\mu_1(1) = \mu_1(2) = 70$ and $\mu_0(2)$ is varied from 5 to 50. In this scenario, the best-path approach believes (based on collected

measurements) that path 2 is the better path (e.g., it may mis-estimate the $\mu_0(2)$ parameter as being less than 20). We vary $\mu_0(2)$ from 5 to 50, in order to see the effect of mis-estimation; hence, the best path approach *over-estimates* this parameter when the real value of $\mu_0(2)$ is less than 20 and *under-estimates* this parameter when the real value of $\mu_0(2)$ is greater than 20. We also consider a very *simple* MP streaming approach, where the load is distributed equally among the two senders in a round-robin manner (i.e., odd-numbered packets are sent along path 1 while even-numbered packets are sent along path 2).

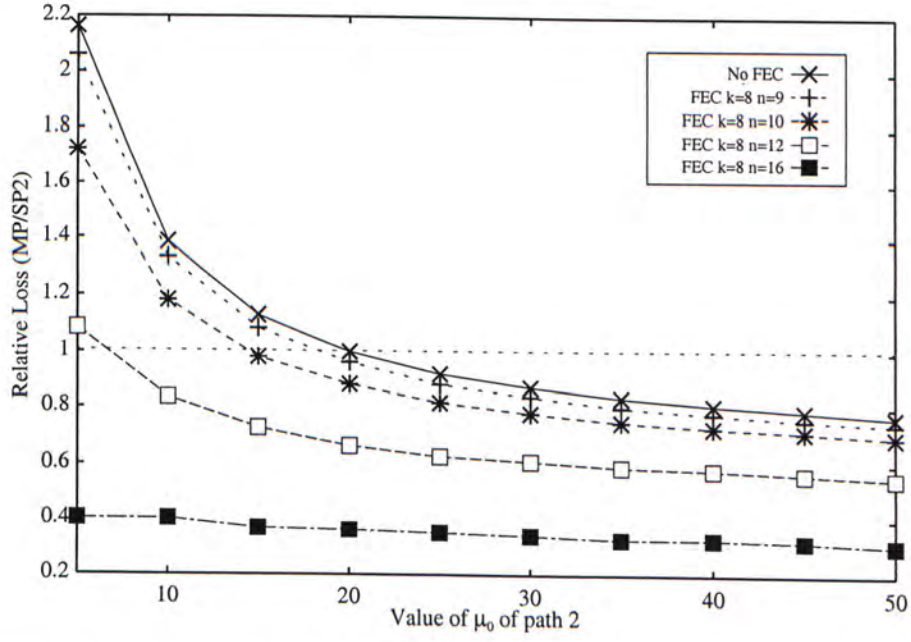


Figure 7.5: Relative loss of dual-path vs. single path when we vary $\mu_0(2)$ and FEC group size.

Figure 7.5 illustrates the *relative* loss rate (using several different FEC schemes) of the two approaches, which is defined as the data loss rate of dual-path streaming divided by the data loss rate of best-path streaming. Hence, a relative loss of less than 1 implies that the simple dual-path approach is doing better than the best path approach. In this figure, we observe that simple dual-path (round-robin) streaming does quite well compared to best-path streaming, even when there is significant differences in loss characteristics between the two paths. Of

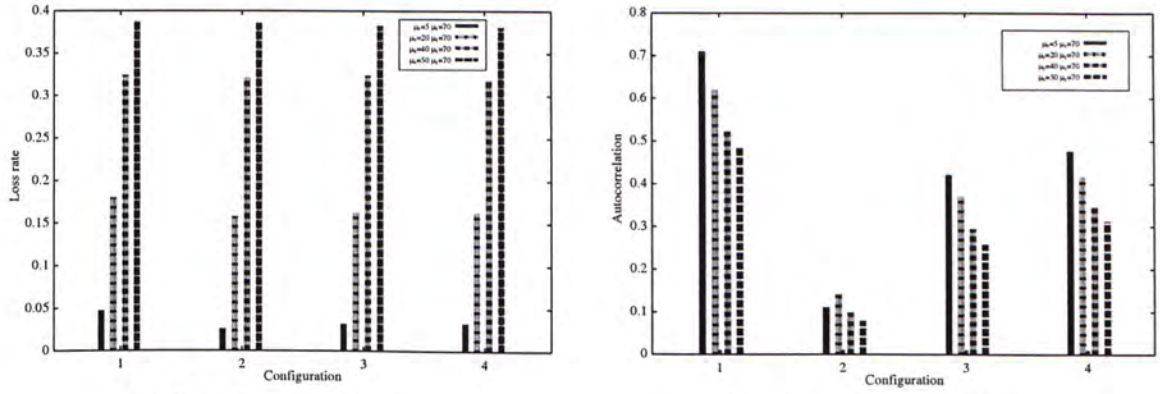
course, in cases where the best path has much better loss characteristics and with relatively little redundant information, the best-path approach has a lower data loss rate. However, we note that the best-path approach would require relatively accurate estimation of the path characteristics, which may be non-trivial especially as network conditions change. Hence, we believe that the MP approach is more robust as compared to best-path streaming.

7.2.7 Effects of Shared Points of Congestion on Various Performance Metrics

In this experiment, we study the effects of *shared points-of-congestion*, between the paths used by the different senders, on various performance measures. Senders S_1 and S_2 share the same point-of-congestion, which we can characterize by a Gilbert model (as defined in chapter 6). Sender S_3 uses a path which does not share a point of congestion with S_1 and S_2 (as before, this path is characterized by a Gilbert model). All the application settings remain the same, and we consider the following four configurations.

- **Configuration 1:** Sender 1 is the only one streaming the data.
- **Configuration 2:** Senders 1 and 3 stream the data in a round-robin manner, i.e., each transmits at a rate of 60 data packets/second.
- **Configuration 3:** Senders 1, 2, and 3 stream the data in a round-robin manner, i.e., each transmits at a rate of 40 data packets/second.
- **Configuration 4:** Senders 1, 2, and 3 stream the data, but senders 1 and 2 transmit at a rate of 20 data packets/second while sender 3 transmits at a rate of 80 data packets/second.

Figure 7.6 illustrates the data loss rate and lag1-autocorrelation for above configurations, when FEC is used, with $(n = 10, k = 8)$. Moreover, we vary the



(a) Data Loss Rate (*b*)Lag1-autocorrelation

Figure 7.6: Effects of shared points-of-congestion on data loss rate and lag-1 autocorrelation with FEC ($n = 10, k = 8$).

$\mu_0(1), \mu_1(1), \mu_0(3)$, and $\mu_1(3)$ parameters (as described in the figure). From this figure, we observe the following.

- MP streaming (configuration 2, 3, and 4) has a lower data loss rate as compared to SP streaming (configuration 1).
- Detecting shared points of congestion is important, as including a greater number of paths in a transmission (under such conditions) may adversely affect the data loss rate. For example, equally splitting the workload among senders 1 and 3 (configuration 2) achieves a lower data loss rate than equally splitting the workload among senders 1, 2, and 3 (configuration 3). This occurs because senders 1 and 2 share the same point of congestion and with configuration 3 we are actually sending a greater fraction of the workload through this shared point of congestion. This agrees with intuition, as in this section we are effectively modeling a shared point of congestion as a single path/bottleneck, i.e., configuration 3 effectively corresponds to a configuration with two senders and an unequal split of workload between them.

- Of course, shared points of congestion adversely affect the lag-1 autocorrelation metric. For example, configuration 3 has a higher lag-1 autocorrelation than configuration 2. Again, the explanation given in the preceding point applies here as well.

7.3 Simulation Model Based Evaluation

In this section, we evaluate the performance of SP streaming vs. MP streaming using the NS-2 [22] simulator. NS-2 is a packet level simulator which allows us to study the performance measures (as defined in Section 6) under more realistic traffic and Internet protocols (such as UDP).

7.3.1 Simulation Setup

As in the previous section, we consider at most three senders (S_1, S_2 and S_3) and one receiver C. Figure 7.7 illustrates our simulation topology. Each sender

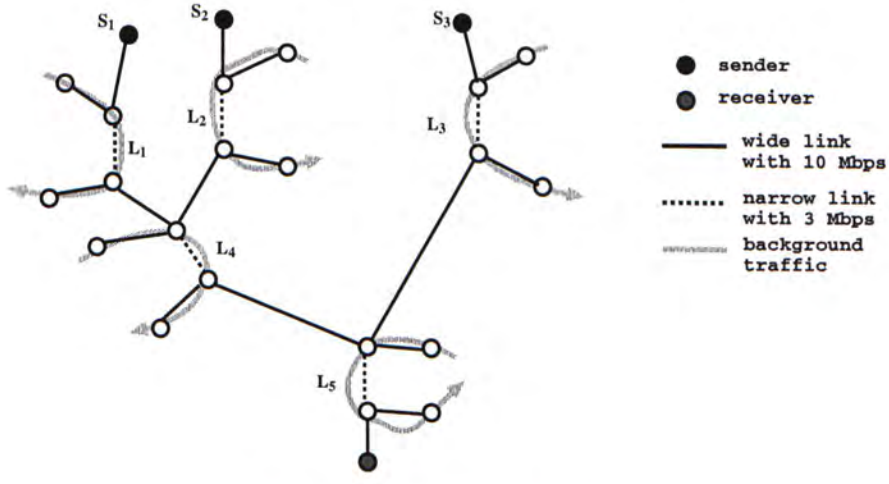


Figure 7.7: Simulation Topology.

transmits the video data, at a constant rate, to the receiver C using the UDP protocol, with packet sizes of 1400 bytes. The data traffic goes through two types of links: (1) wide/higher capacity links (represented by solid lines) and (2)

narrow/lower capacity links (represented by dotted lines). Each wide link has a bandwidth of 10 Mbps while the bandwidth of a narrow link is 3 Mbps. Each link has a different propagation delay and the propagation delay is generated using an exponential random variable with a mean of 200 ms. The streaming application has a sending rate of 1.5 Mbps which consumes 50% of the bandwidth of a narrow link. The actual sending rate of each sender is a function of the traffic load distribution. Unless stated otherwise, an equal distribution is used, e.g., for MP streaming with three senders (sending data in a round-robin manner), the sending rate of each sender is 0.5 Mbps. Background traffic (represented by grey arrows) is introduced at different narrow links. The background traffic is generated using exponential on/off sources. The average "on" time plus the average "off" time of these on/off sources is equal to 1 second. During the "on" times, the background source generates UDP traffic with a constant rate of 3 Mbps, which can saturate the capacity of the traversed narrow links. In the following experiments we vary the amount of "on" time within an average of 1 second period. For example, a background traffic rate of 1.8 Mbps represents an average "on" time of 0.6 seconds for an average of 1 second on/off period. There are three possible sets of background traffic locations. One set of local background traffic occurs on the narrow links L_i where $i = 1, 2, 3$. This background traffic competes with the corresponding sender S_i $(i = 1, 2, 3)$ for the bandwidth resources of the narrow links L_1, L_2, and L_3, respectively. The second set of background traffic occurs on the narrow link L_4. This background traffic competes with senders S_1 *and* S_2 for the bandwidth resource of the narrow link L_4. The third set of background traffic occurs on the narrow link L_5. This background traffic competes with *all* three senders for the bandwidth resource of the narrow link L_5. Unless stated otherwise, SP streaming is done from sender 1 and dual-path streaming is done from senders 1 and 3.

7.3.2 Data Loss Rate

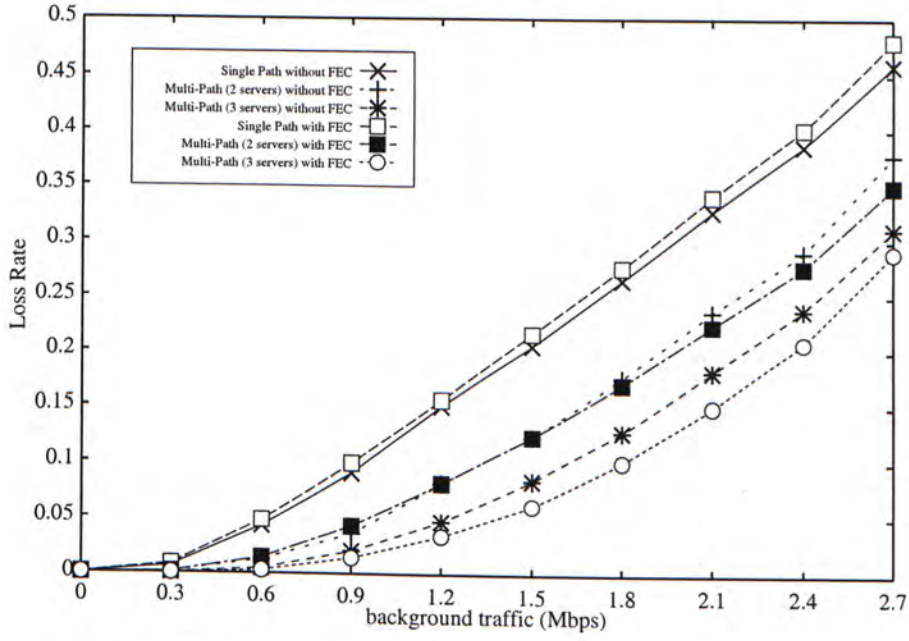


Figure 7.8: Loss rate with FEC parameters $n = 10$ and $k = 8$.

Figure 7.8 illustrates the data loss rates for SP and MP streaming. In this simulation, we vary the average background traffic through the narrow links L_1, L_2, and L_3 from 0 Mbps to 2.7 Mbps. (Note that the senders do not share points of congestion in this case.) From this figure, we observe the following. Firstly, MP streaming can achieve a significant reduction in the data loss rate as compared to SP streaming. Secondly, employment of FEC may actually increase the data packet loss rate; for example, the data loss rate of SP streaming with FEC is a bit higher than the data loss rate of SP without FEC. Thirdly, the improvements in the data loss rate achieved through the use of MP streaming *without* FEC is higher than that achieved through the use of FEC by adding it to SP streaming. This is potentially due to the fact that the use of FEC (with SP streaming) introduces additional traffic into the (already) congested network and hence results in higher data losses. On the other hand, the use of MP streaming achieves a significant reduction in data loss rate without introduction of additional network traffic.

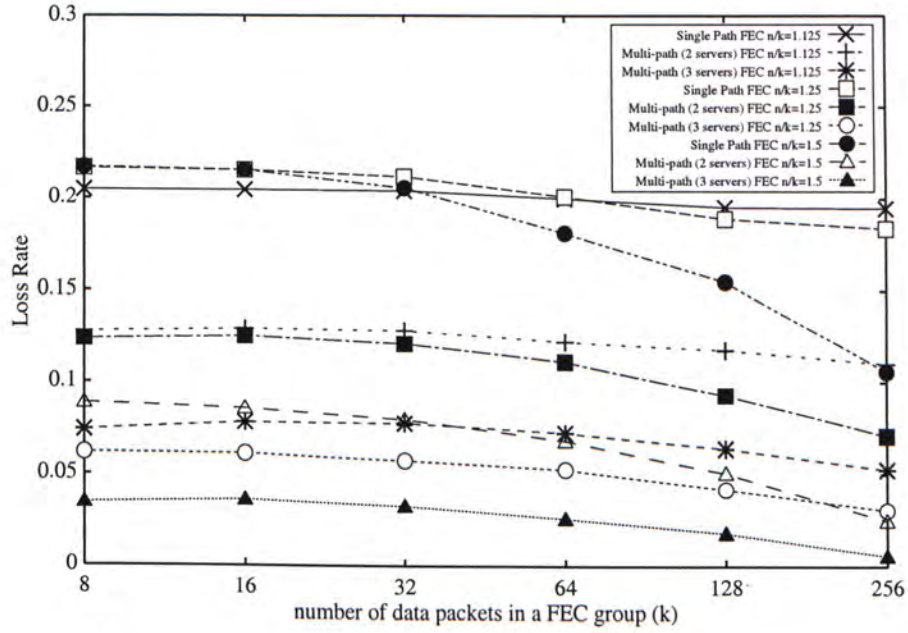


Figure 7.9: Loss rate as a function of n/k ratio and k

7.3.3 Data Loss Rate as a function of FEC parameters

In this experiment, we study the effects of FEC parameters on the data loss rate. Again, we vary the FEC parameters as in Section 7.2. Figure 7.9 illustrates the data loss rate when a background traffic of 1.5 Mbps is used on each of the narrow links L_1, L_2, and L_3. We observe that:

- Increasing the degree of redundancy under SP streaming may not necessarily reduce the data loss rate, one reason being that introducing additional traffic (due to higher degree of redundancy) into an already congested network may result in higher packet loss rates. Hence, MP streaming may have a higher chance of decreasing the data loss rate with higher degrees of redundancy, i.e., with less traffic being introduced per path.

- MP streaming can significantly reduce data loss rate as compared to SP streaming.

In summary, we observe that increasing the amount of redundancy (by increasing the n/k ratio) or increasing the FEC group size (and hence potentially suffering

higher latency at the receiver with a need for larger buffer sizes) may not result in significant reduction in data loss rate, for either SP or MP streaming. On the other hand, taking advantage of multiple independent paths, can reduce the data loss rate significantly.

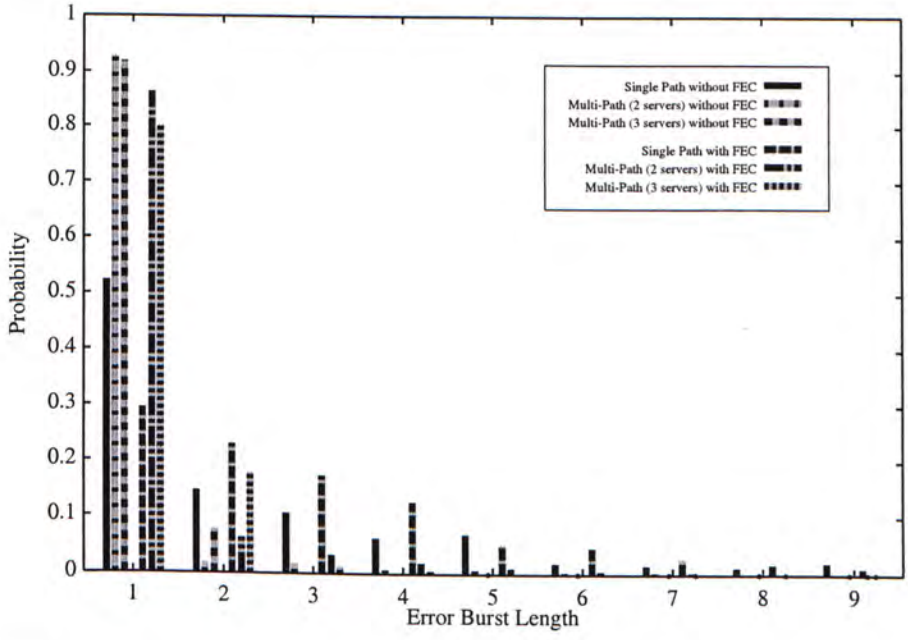


Figure 7.10: Conditional probability mass function for error burst length.

7.3.4 Conditional Error Burst Length

In this experiment, we compare the conditional burst length distribution, conditioned on there being at least one loss, of the SP and MP approaches. In this case a background traffic of 2.4 Mbps is used on each of the narrow links L_1, L_2 and L_3. The conditional probability mass function[3] of error burst length is given in Figure 7.10, where we observe that MP streaming has a stochastically smaller data packet burst length than SP streaming.

[3]As in Section 7.2 we illustrate the probability mass function rather than the probability distribution function, as we believe it depicts the results of the experiment better.

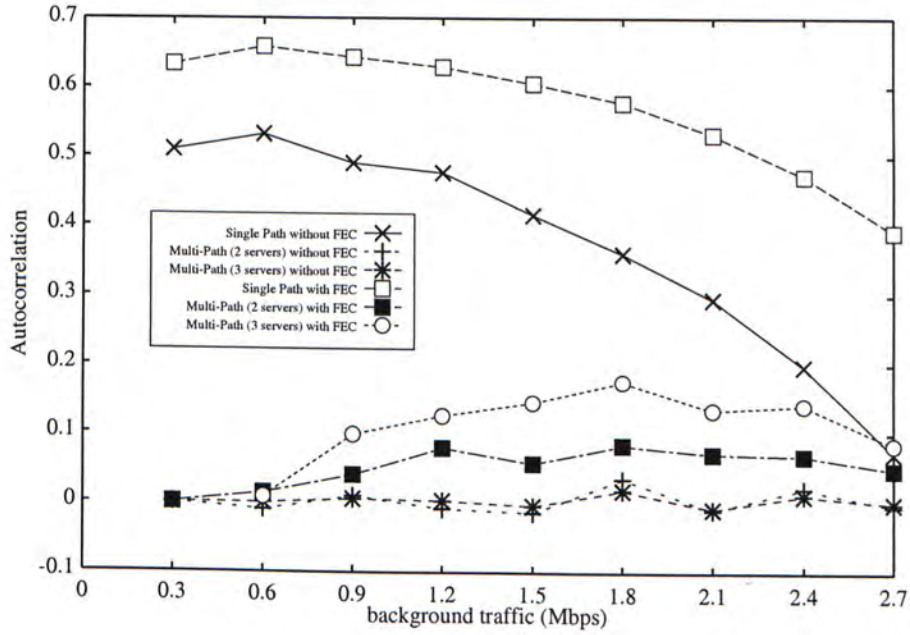


Figure 7.11: Lag-1 autocorrelation.

7.3.5 Lag-1 Autocorrelation

In this experiment, we study lag-1 autocorrelation of packet losses for both SP and MP streaming. Figure 7.11 illustrates the lag-1 autocorrelation as we vary the background traffic on the narrow links L_1, L_2 and L_3. We observe the following:

- Without use of FEC, the MP lag-1 autocorrelation is close to zero (as derived in Section 6), i.e., the losses appear nearly uncorrelated when streaming over multiple independent paths. On the other hand, the correlation of losses with SP streaming can be quite high.

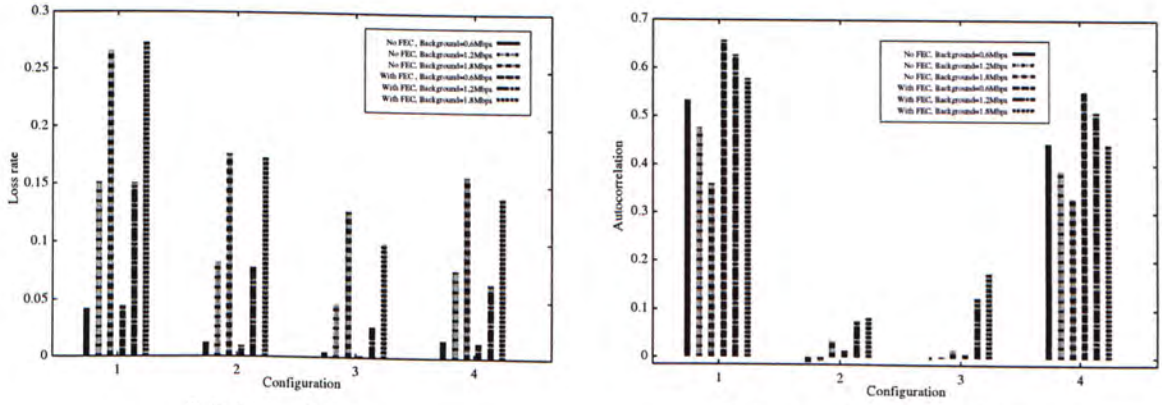
- With use of FEC, lag-1 autocorrelation may increase. We believe that a similar explanation (as given in experiment of section 7.2.4) holds here. However, we still observe that the MP lag-1 autocorrelation is significantly lower than the SP lag-1 autocorrelation (under the same FEC scheme).

Lastly, the decrease in lag-1 autocorrelation as a function of higher background traffic may be counter-intuitive. One explanation may be that the "no losses"

(i.e., the packets that are received successfully) in the resulting stream tend to be more "random" as congestion on the network increases.

7.3.6 Effects of Load Distribution among Senders

In previous experiments, all senders transmitted packets in a round-robin manner and hence the load on all senders (i.e., the amount of data streamed from each sender) was the same. In this experiment, we study the effects of different load distributions on the resulting loss characteristics observed at the receiver. Specifically, we consider the following configurations.



(a) Loss Rate (*b*)Lag-1 Autocorrelation

Figure 7.12: Loss rate and Lag-1 autocorrelation under different load distributions

Configuration 1 Streaming from sender 1 only.

Configuration 2 Equal distribution of load between senders 1 and 3 only.

Configuration 3 Equal distribution of load among all senders.

Configuration 4 Sender 1 streams 1/6 of the data, sender 2 streams 1/6 of the data, and sender 3 streams 2/3 of the data.

Figure 7.12 depicts the data loss rate and the lag-1 autocorrelation of these configurations. In this experiment, equal distribution of load (configuration 3) tends

to achieve a lower data loss rate and lag-1 autocorrelation.

7.3.7 Sensitivity Analysis

In this experiment, we study the relative performance of MP streaming vs. SP streaming when the SP streaming is performed over the *best* of the available paths (please refer to Section 7.2 for a more detailed explanation of "best path" streaming and the motivation for making this comparison). Specifically, we consider a two senders system with only senders S_1 and S_3 transmitting packets. The background traffic on L_1 is fixed at 1.5 Mbps, and the background traffic on L_3 is varied from 0.3 to 2.7 Mbps. In this scenario, the best-path approach believes (based on collected measurements) that the path originating at sender S_3 experiences the least losses. Therefore, the best-path streaming approach always uses the path originating from sender S_3. We also consider a very *simple* MP streaming approach, which streams the data in a round-robin manner from S_1 and S_3.

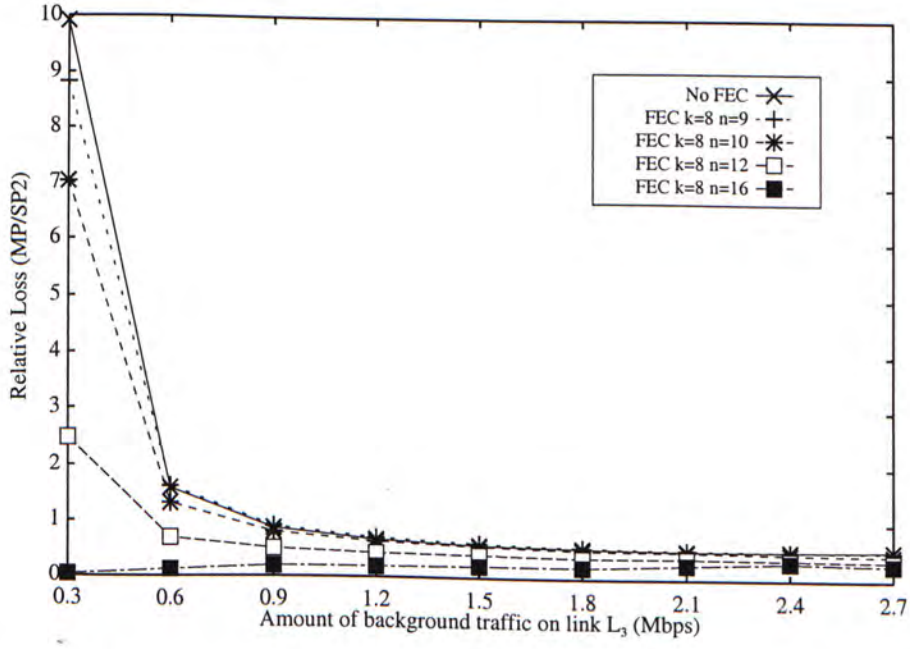


Figure 7.13: Relative Loss Rate when background traffic on link L_3 and FEC group size are varied.

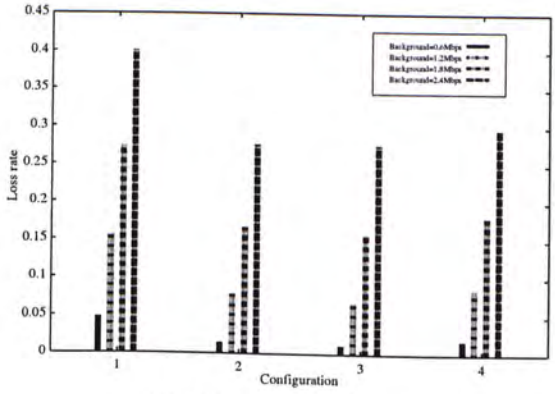
Figure 7.13 illustrates the *relative* loss rate (using several different FEC schemes), which is defined as the data loss rate of dual-path streaming divided by the data loss rate of best-path streaming. Hence, a relative loss rate of less than 1, implies that simple dual-path streaming is more robust as compared to best-path streaming. As in Section 7.2, we observe that simple dual-path (round-robin) streaming does quite well compared to best-path streaming, even when there is significant differences in loss characteristics between the two paths. Of course, in cases where the best path has much better loss characteristics and with relatively little redundant information, the best-path approach has a lower data loss rate. Hence, we believe that the MP approach is more robust as compared to best-path streaming.

7.3.8 Effects of Shared Points of Congestion on Various Performance Metrics

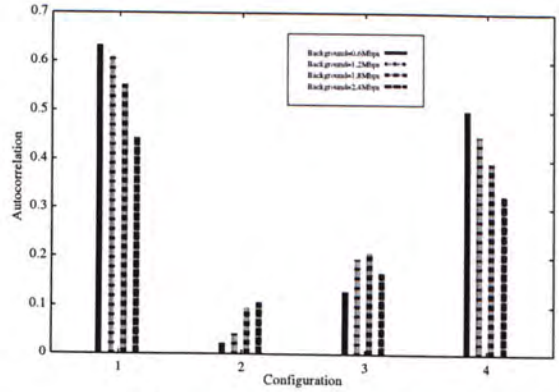
In this experiment, we study the effects of *shared points-of-congestion*, between the paths used by the different senders, on various performance measures. Here, the background traffic is sent through the narrow links L_3 and L_4. Note that, having background traffic on L_4 implies that senders 1 and 2 share the same point-of-congestion. Again, we consider the four configurations described in experiment of section 7.3.6 above.

Figure 7.14 illustrates the data loss rate and lag-1 autocorrelation for these configurations, when FEC is used, with $(n = 10, k = 8)$. Moreover, we vary the background traffic on the two narrow links L_3 and L_4 among the following values: 0.6 Mbps, 1.2 Mbps, 1.8 Mbps, and 2.4 Mbps. From this figure, we observe the following:

- MP streaming (configurations 2, 3, 4) has a lower data loss rate as compared to SP streaming (configuration 1).
- Detecting shared points of congestion is important, as including a greater



(a) Data Loss Rate



(*b*)Lag1-autocorrelation

Figure 7.14: Effects of shared points-of-congestion on data loss rate and lag-1 autocorrelation with FEC ($n = 10, k = 8$).

number of paths/senders (under such conditions) in the transmission may adversely affect the data loss rate.

- Shared points of congestion adversely affect the lag-1 autocorrelation metric. For example, configuration 3 has a higher lag-1 autocorrelation than configuration 2.

Chapter 8

Conclusion

In this thesis, we investigated the potential benefits of the multi-path streaming approach. The main advantage of the multi-path streaming approach is that it does not require support from lower layers. Hence, this approach is scalable and readily deployable. We focus on the delivery of pre-recorded continuous media over the best-effect wide-area networks, and we use the Gilbert model to analyse the multi-path streaming approach to provide a fundamental understanding of the benefits.

Our results on analysis and simulations indicate that multi-path streaming has better loss characteristics than single-path streaming, both with or without the use of erasure code. The better loss characteristics of the multi-path streaming should result in improvement of visual quality of the received continuous media.

We also discuss issues on end-point adaptations. Results show that paths with shared point-of-congestion reduce the performance and should be avoided. Besides, simple round-robin load distribution is fairly robust.

We believe these results are encouraging. Future works on multi-path streaming includes further study on dynamic adaptations and real Internet experiments.

Bibliography

[1] R. L. Carter and M. Crovella, Server selection using dynamic path characterization in wide-area networks, in *INFOCOM (3)*, pages 1014–1021, 1997.

[2] N. F. Maxemchuk, Dispersity routing, in *the IEEE International Conference on Communications*, San Francisco, California, June 1975.

[3] N. F. Maxemchuk, Dispersity routing in high-speed networks, in *the Workshop on Very High Speed Networks*, Greenbelt, Maryland, March 1991.

[4] N. F. Maxemchuk, Dispersity routing in an atm network, in *the IEEE INFOCOM*, San Francisco, California, March/April 1993.

[5] H. Chu and K. Nahrstedt, Dynamic multi-path communication for video traffic, in *the Hawiian International Conference on System Science*, Hawaii, January, 1997.

[6] J.-C. Chen and S.-H. Chan, Multipath routing for video unicast over bandwidth-limited networks, in *the IEEE Globecom*, San Antonio, Texas, November 2001.

[7] Y. J. Liang, E. G. Steinbach, and B. Girod, Real-time voice communication over the internet using packet path diversity, in *the ACM Multimedia Conference*, Ottawa, Canada, September/October 2001.

[8] Y. J. Liang, E. G. Steinbach, and B. Girod, Multi-stream voice over ip using packet path diversity, in *the IEEE Fourth Workshop on Multimedia Signal Processing*, Cannes, France, October 2001.

[9] T. Nguyen and A. Zakhor, Distributed video streaming over internet, in *the SPIE Conference on Multimedia Computing and Networking*, San Jose, California, January 2002.

[10] T. Nguyen and A. Zakhor, Distributed video streaming with forward error correction, in *the International Packetvideo Workshop*, Pittsburg, Pennsylvania, April 2002.

[11] J. Bolot, Characterizing end-to-end packet delay and loss in the internet, 1993.

[12] V. Paxson, IEEE/ACM Transactions on Networking **7**, 277 (1999).

[13] J. Bolot, S. Fosse-Parisis, and D. Towsley, Adaptive fec-based error control for interactive audio in the internet.

[14] M. Yajnik, J. Kurose, and D. Towsley, Packet loss correlation in the MBone multicast networ experimental measurements and markov chain models, Technical Report UM-CS-1995-115, 1995.

[15] P. Morse, *Queues, Inventories, and Maintenance*, John Wiley, 1958.

[16] Rizzo, ACMCCR: Computer Communication Review **27** (1997).

[17] B. Ahlgren, M. Bjo, and r Melander, Network probing using packet trains, 1999.

[18] D. Rubenstein, J. Kurose, and D. Towsley, Detecting shared congestion of flows via end-to-end measurement, in *the ACM Sigmetrics Conference*, Santa Clara, California, June, 2002.

[19] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, Adaptive FEC-Based Error Control for Internet Telephony, in *INFOCOM*, 1999.

[20] S. Ross, *Stochastic Processes*, John Wiley, 1996.

[21] D. L. Gall, Communications of the ACM (April 1991).

[22] http://www.isi.edu/nsnam/ns/, *The Network Simulator - ns-2.*